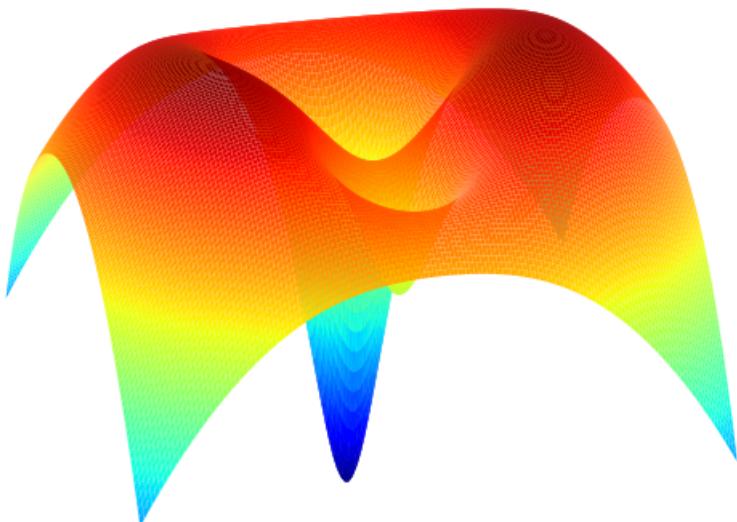


Mecánica Celeste

Teoría, algoritmos y problemas



Jorge I. Zuluaga

Profesor titular de Física y Astronomía

Instituto de Física, Facultad de Ciencias Exactas y Naturales
Universidad de Antioquia

3 de marzo de 2020

BORRADOR

Índice general

| | | |
|-----------|---|-----------|
| 1. | Prefacio | 7 |
| 1.1. | ¿Otro libro de mecánica celeste? | 8 |
| 1.2. | Mecánica celeste y mecánica analítica | 9 |
| 1.3. | Mecánica celeste en la era de la información | 10 |
| 1.4. | Mecánica celeste en Python | 11 |
| 1.5. | Mecánica celeste con SPICE | 12 |
| 1.6. | ¿Qué hace distinto a este libro?: un decálogo | 13 |
| 2. | Agradecimientos | 17 |
| 3. | Introducción | 19 |
| 3.1. | ¿Cómo se organiza este libro? | 19 |
| 3.2. | ¿Cómo usar este libro? | 24 |
| 3.3. | Mecánica celeste en <i>libretas</i> | 25 |
| 3.3.1. | Instalación de las libretas | 26 |
| 3.4. | Idioma y Notación | 27 |
| 3.4.1. | Palabras extranjeras y guía de pronunciación | 27 |
| 3.4.2. | Siglos y décadas | 27 |
| 3.4.3. | Notación matemática | 28 |
| 3.5. | Elementos no textuales | 28 |
| 3.5.1. | Cajas de texto | 28 |
| 3.5.2. | Algoritmos | 30 |
| 3.6. | Figuras interactivas y animaciones | 35 |
| 4. | Fundamentos matemáticos | 37 |
| 4.1. | Vectores y cálculo | 37 |
| 4.1.1. | Conjunto, tuplas y vectores | 38 |
| 4.1.2. | Sistemas de coordenadas | 44 |
| 4.1.3. | Funciones | 46 |
| 4.1.4. | Derivadas | 48 |
| 4.1.5. | Funciones homogéneas | 50 |
| 4.1.6. | Derivada vectorial | 51 |
| 4.1.7. | Integrales | 53 |
| 4.1.8. | Integrales vectoriales | 54 |
| 4.1.9. | Ecuaciones diferenciales | 56 |
| 4.1.10. | Funcionales y cálculo de variaciones | 62 |

| | | |
|---------|--|------------|
| 4.1.11. | Gráficos interactivos | 69 |
| 4.1.12. | Series infinitas | 69 |
| 4.2. | Curvas cónicas | 82 |
| 4.2.1. | Definición geométrica | 83 |
| 4.2.2. | Del nombre al álgebra | 84 |
| 4.2.3. | Directriz de las cónicas | 87 |
| 4.2.4. | Síntesis geométrica | 89 |
| 4.2.5. | Descripción algebraica | 90 |
| 4.2.6. | Ecuación respecto al centro | 90 |
| 4.2.7. | Eje mayor y menor de la elipse | 91 |
| 4.2.8. | Parámetros de la hipérbola | 92 |
| 4.2.9. | Rotación de las cónicas en el plano | 94 |
| 4.2.10. | Ecuación general de las cónicas | 96 |
| 4.2.11. | Gráfico de una cónica rotada en el plano | 97 |
| 4.2.12. | Síntesis algebraica | 102 |
| 4.2.13. | Cónicas en coordenadas cilíndricas | 103 |
| 4.2.14. | Anomalías | 106 |
| 4.2.15. | Área de las cónicas | 109 |
| 4.2.16. | Cónicas en el espacio | 113 |
| 4.2.17. | Ángulos de Euler | 114 |
| 4.2.18. | Matrices de rotación generales | 115 |
| 4.2.19. | Gráfico de una cónica rotada en el espacio | 117 |
| 4.2.20. | Elementos orbitales | 119 |
| 4.3. | Problemas seleccionados | 123 |
| | Bibliografía | 127 |

Índice de figuras

| | | |
|------|---|----|
| 1.1. | Imagen procesada de Arrokoth, el objeto transneptuniano sobrevolado por la sonda New Horizons el 1 de enero 2019 (crédito: NASA/Johns Hopkins University Applied Physics Laboratory/- Southwest Research Institute/Roman Tkachenko.) | 8 |
| 1.2. | Figura correspondiente al código 1.1. | 12 |
| 3.1. | Ilustración esquemática del <i>teorema de Danelin</i> | 30 |
| 3.2. | Retrato de Johanes Kepler, copia de un original de 1610 de pintor desconocido y que se conserva en el monasterio Benedictino de Kremsmünster (Alemania). | 31 |
| 3.3. | Figura correspondiente al código 3.3. | 35 |
| 3.4. | Gráfico de las funciones trigonométricas básicas, en el intervalo de interés (gráfico generado). | 35 |
| 4.1. | Definición geométrica de vector espacial y de sus operaciones básicas (suma, resta y multiplicación por un escalar). Aunque la resta de $\vec{A} - \vec{B}$ es un caso particular de la suma, es importante aquí familiarizarse con la dirección que tiene este vector (va de la cabeza del sustraendo \vec{B} a la del minuendo \vec{A}). | 39 |
| 4.2. | Definción de los sistemas de coordenadas usadas en este texto | 44 |
| 4.3. | Figura correspondiente al código 4.3. Solución aproximada de la ecuación diferencial $d^2F/dt^2 = -kF$ con $k=1.5$ | 61 |
| 4.4. | El área bajo una curva es un funcional, en tanto depende de la función que represente la curva, $f(t)$ o $f_0(t)$ Se conoce como una variación δf a la diferencia entre dos funciones cercanas, parametrizada a través de un número real ϵ y una función plantilla (panel inferior.) En términos de variaciones el valor de cualquier función vecina a una función de referencia f_0 se puede calcular, en un intervalo de interés, como $f(t) = f_0(t) + \epsilon\eta(t)$ | 63 |
| 4.5. | Figura correspondiente al código 4.6. La curva continua indica una aproximación numérica al camino más corto entre los puntos $(0, 0)$ y $(0, \pi)$ del plano euclíadiano, encontrada al minimizar el funcional longitud de arco y usando como función de prueba $f_0 = (t/\pi)^n$ (línea punteada) y como función plantilla $\epsilon(t) = \sin t$. El valor de ϵ que corresponde a la solución se muestra en la etiqueta. Para comparación se muestra (línea rayada) la solución exacta, que corresponde a una línea recta. | 69 |

| | | |
|-------|---|-----|
| 4.6. | Figura correspondiente al código 4.11. | 82 |
| 4.7. | Definición geométrica original de las <i>curvas cónicas</i> | 83 |
| 4.8. | Definición con <i>áreas aplicadas</i> de las <i>curvas cónicas</i> y el origen de sus nombres. | 84 |
| 4.9. | Figura correspondiente al código 4.12. | 86 |
| 4.10. | Definición de las cónicas usando la recta directriz y el foco. | 87 |
| 4.11. | Parámetros geométricos de la elipse referidos al apside O, el foco F y el centro C: <i>a</i> semieje mayor, <i>b</i> semieje menor, <i>p semilatus rectum</i> , <i>e</i> excentricidad, <i>c</i> distancia foco-centro. | 92 |
| 4.12. | Parámetros geométricos de la hipérbola referidos al apside O, el foco F y el vértice C: <i>a</i> distancia al vértice (llamado con frecuencia también semieje mayor aunque en la hipérbola no hay tal), β pendiente de la hipérbola, <i>p semilatus rectum</i> , <i>e</i> excentricidad, ψ angulo de semiapertura. | 93 |
| 4.13. | Figura correspondiente al código 4.14. | 99 |
| 4.14. | Figura correspondiente al código 4.16. | 100 |
| 4.15. | Derivación de la ecuación de la cónica en coordenadas cilíndricas referidas al Foco. En la figura el ángulo <i>f</i> es la <i>anomalía verdadera</i> | 103 |
| 4.16. | Figura correspondiente al código 4.18. | 105 |
| 4.17. | Figura correspondiente al código 4.20. | 107 |
| 4.18. | Definición de la anomalía excéntrica <i>E</i> y el método asociada a ella para determinar la posición de los puntos sobre una elipse. | 108 |
| 4.19. | Anomalía verdadera <i>f</i> como función de la anomalía excéntrica <i>E</i> para una elipse. La línea rayada corresponde a la aproximación $f \approx \sqrt{(1+e)/(1-e)}E$ que es valida en el caso $f \ll 1$ | 109 |
| 4.20. | Anomalía excéntrica <i>F</i> como función de la anomalía verdadera <i>f</i> para una hipérbola. La línea punteada corresponde a la aproximación $F \approx \sqrt{(e-1)/(e+1)}f$ | 110 |
| 4.21. | Construcción geométrica usada aquí para deducir la relación entre las anomalías <i>f</i> , <i>E</i> y <i>F</i> y el área del sector de cónica (región sombreada). | 111 |
| 4.22. | Secuencia de rotaciones que permiten pasar del sistema natural de ejes de la cónica $x - y - z$ a un sistema con una orientación arbitraria $x''' - y''' - z'''$ | 114 |
| 4.23. | Figura correspondiente al código 4.23. | 118 |
| 4.24. | Figura correspondiente al código 4.25. | 122 |
| 4.25. | Una elipse con algunos puntos resaltados. | 123 |
| 4.26. | Definición de las coordenadas de latitud y longitud sobre la Tierra | 126 |

Capítulo 1

Prefacio

En 2019 celebramos el centenario de la histórica observación de un eclipse total de Sol, liderada por *Sir Arthur Eddington* y que permitió la primera confirmación experimental de las predicciones de la teoría general de la relatividad. El primer día de ese mismo año, una nave espacial, la sonda **New Horizons**, sobrevoló el cuerpo astronómico más remoto fotografiado por nuestra especie, el objeto transneptuniano (486958) **Arrokoth**; la misma sonda, cinco años antes, había pasado “rozando” la superficie de Plutón, enviándonos imágenes inesperadas de un mundo sorprendente. Muy lejos de allí, y también en 2019, dos naves espaciales, una japonesa, la sonda **Hayabusa 2** y la otra estadounidense, **OSIRIS-REx**, transmitieron imágenes impactantes desde la superficie de dos pequeños asteroides cercanos a la Tierra, cuerpos que visitaron con el objeto de traer muestras a la Tierra. Lo que aprendamos de esas muestras podría ayudarnos a evitar un impacto catastrófico futuro.

Todas estas hazañas de exploración y conocimiento fueron posibles, entre otras, gracias a la **Mecánica Celeste**. Esta disciplina científica, combinación asombrosa de astronomía, física y matemáticas, comenzó con el trabajo teórico pionero de *Johannes Kepler* a principios de los 1600¹; se estableció con la obra cumbre de *Sir Isaac Newton*, los *Principios Matemáticos de la Filosofía Natural* [6], publicada a finales de los 1600; y alcanzó su apogeo entre los 1700 y los 1800 con los trabajos de matemáticos y astrónomos como *Edmund Halley*, *Leonhard Euler*, *Pierre-Simon Laplace*, *Joseph-Louis Lagrange*, *William Rowan Hamilton* y *Henri Poincaré* (entre muchos otros que mencionaremos en este libro).

Este libro presenta una visión panorámica de la **mecánica celeste** y en general de la **mecánica analítica** o **mecánica clásica**, que se desarrolló de forma paralela a la primera, inspirada, en muchos casos, por sus problemas. El texto está dirigido especialmente a quienes, por su formación o trabajo, están interesados en la aplicación de la mecánica celeste en astronomía o en ingeniería aeroespacial. Su extensión, énfasis y nivel de profundidad lo hace especialmente adecuado para **estudiantes de pregrado** (licenciatura o bachillerato, dependiendo del país) de

¹En la [Capítulo 3](#) haremos claridad sobre la nomenclatura usada en el libro para referirnos a los siglos y decenios.



Figura 1.1: Imagen procesada de Arrokoth, el objeto transneptuniano sobrevolado por la sonda New Horizons el 1 de enero 2019 (crédito: NASA/Johns Hopkins University Applied Physics Laboratory/Southwest Research Institute/Roman Tkachenko.)

cualquier programa científico o técnico, especialmente astronomía, física o ingeniería aeroespacial. Su enfoque computacional, lo podría hacer, además, útil como material de referencia para profesionales de estas disciplinas.

1.1. ¿Otro libro de mecánica celeste?

Al escribir este libro, no pretendo hacer un compendio exhaustivo de los problemas de la Mecánica Celeste, que, durante más de 400 años de historia se ha convertido en una disciplina científica basta y en constante desarrollo.

Muchos textos en la materia han sido escritos desde los tiempos de Newton, la mayoría en las últimas décadas. Algunos presentan detallados y rigurosos desarrollos matemáticos. Otros están orientados específicamente al Sistema Solar o al movimiento de satélites y vehículos espaciales. Muchos más son buenos libros de texto, la mayoría dirigidos a estudiantes de posgrado (la mecánica celeste es considerada una línea de profundización, tanto en física como en astronomía.) También se han escrito algunos libros divulgativos y al alcance de aficionados.

La bibliografía de este libro recoge apenas una muestra de referencias en la materia, que serán citados a lo largo de sus capítulos, y que, de antemano, invito a los lectores a explorar con curiosidad para no quedarse con la punta de el inmenso *iceberg* que apenas alcanzará a asomarse en estas páginas.

Siendo este el caso ¿para qué escribir un libro más de mecánica celeste? Existen dos razones fundamentales que me motivaron a emprender esta aventura.

La primera es que, como mencione antes, la mayoría de los libros de mecánica celeste están dirigidos a estudiantes con una formación media o avanzada en matemáticas, mecánica newtoniana y mecánica analítica. Como se acostumbra decir, tienen un nivel de posgrado. En contraste, el número de textos al “alcance” de estudiantes de los primeros años universitarios, no es muy grande. Escribo este libro para contribuir a Enriquecer precisamente ese “nicho”.

Podría argumentarse que la mecánica celeste, como aplicación específica de la mecánica, es un tema especializado y de allí que sus textos estén dirigidos a estudiantes más avanzados. Sin embargo, la importancia de esta disciplina en la historia de la astronomía y de la física, así como su potencial para describir fenómenos fascinantes, desde el movimiento de planetas y naves espaciales, hasta la colisión de agujeros negros, hace de la mecánica celeste un medio educativo excelente para introducir conceptos teóricos en física y astronomía, que, sin un contexto y motivación apropiado, son difíciles de digerir.

Un buen libro de mecánica celeste o mecánica analítica, sin importar su nivel, debería poder ser estudiado por cualquier estudiante, incluso de pregrado. Esa ha sido la premisa en muchos centros académicos. Pero la realidad es más compleja. Como cualquier profesor sensible sabe, para valorar realmente los logros intelectuales del pasado, entender las motivaciones que llevaron a los padres de una disciplina a introducir hipótesis o formular las leyes de la misma, se necesita experiencia académica. Experiencia que la mayoría de los estudiantes de pregrado no tienen. No es solo un problema de nivel matemático, es también un problema de falta de exposición a la ciencia.

Este libro, pretende ser un buen *primer* libro de mecánica celeste, pero también de mecánica analítica, como explicaremos más adelante. Un primer escalón para abordar, ya con experiencia, libros más avanzados.

1.2. Mecánica celeste y mecánica analítica

La segunda razón, y la original para mi como profesor del **pregrado de Astronomía en la Universidad de Antioquia**, fue la necesidad de escribir un texto de mecánica celeste que permitiera además una formación en los principios y métodos de la mecánica analítica (mecánica teórica o mecánica clásica). Esos principios y métodos son instrumentales en la formulación de la mecánica cuántica y lo son además en versiones modernas de otras áreas de la física clásica, como la relatividad, la electrodinámica e incluso la óptica. La mecánica analítica es indispensable entonces en la formación de cualquier estudiante de ciencias físicas.

En la inmensa mayoría de los textos clásicos de mecánica celeste, los resultados se derivan usando, casi exclusivamente, los métodos de lo que llamaremos aquí el **formalismo vectorial o geométrico de la mecánica**. En este formalismo (originalmente introducido por Newton y desarrollado posteriormente por Euler) las fuerzas juegan el papel central en la descripción de la dinámica (*dime cuánto te halan y te diré cómo te mueves.*)

Desde los trabajos pioneros de matemáticos y “físicos” de los 1700 y 1800, tales como *Alambert*, *Lagrange*, *Hamilton* y *Jacobi*, se hizo evidente que algunos problemas complejos de mecánica celeste podían abordarse usando un **formalismo analítico o escalar de la mecánica**. En este formalismo, los sistemas se describen usando *funciones* tales como el *Lagrangiano* o el *Hamiltoniano*, que contienen toda la información relevante del sistema, sus restricciones y simetrías (*dime cuál es tu hamiltoniano y no solo te diré para dónde vas sino también cómo eres.*)

Un caso ilustrativo, muy popular y reciente, de como el formalismo analítico de la mecánica es aplicado hoy, de forma generalizada, en mecánica celeste, es la “predicción” de un nuevo planeta en el Sistema Solar, más allá del cinturón de Kuiper, cuya existencia, a la fecha, no se ha confirmado, ni rechazado [3]. Este

trabajo también es la punta de un inmenso “iceberg” de literatura científica en mecánica celeste en la que el formalismo analítico es protagonista.

Más allá entonces de la necesidad práctica de juntar a la mecánica celeste y a la mecánica analítica en un mismo texto, de modo que sirva a estudiantes de programas académicos como astronomía o ingeniería aeroespacial, este libro presenta este particular “matrimonio” entre dos disciplinas clásicas de la astronomía y la física como lo que es: una relación estrecha entre dos cuerpos de conocimiento inseparables.

1.3. Mecánica celeste en la era de la información

Un ingrediente adicional hace a este libro diferente. Me refiero al enfasis especial que daremos a los algoritmos de la mecánica celeste a través de todo el libro.

Es un hecho reconocido que la complejidad de muchos problemas de mecánica celeste, en particular aquellos con un interés práctico tales como el diseño de trayectorias de vehículos espaciales, la predicción de la posición precisa de asteroides y cometas que pueden amenazar nuestro planeta o la predicción a largo plazo de la posición de los cuerpos del sistema solar y otros sistemas planetarios, ha exigido, casi desde los tiempos de Kepler, el desarrollo y aplicación de métodos numéricos y, más recientemente, su implementación en calculadores y computadores.

En este sentido, la relación de la mecánica celeste con *algoritmos* de toda clase, no es comparable con la relación, principalmente utilitaria, que tienen la mayoría de las áreas de la física con la computación. Podría decirse, que hoy, es casi impensable saber de mecánica celeste, sin estar familiarizado también con sus algoritmos.

Pensando en esto, todo el contenido del libro ha sido elaborado usando *libretas* o *notebooks* del Proyecto Jupyter². Estas libretas pueden ser obtenidas y usadas por el lector para interactuar con y modificar los algoritmos (el material electrónico está disponible en el sitio en línea³ del libro). Estos medios tecnológicos permiten además aprovechar gráficos interactivos y animaciones para entender mejor conceptos que pueden ser difíciles.

En la versión impresa, los algoritmos se presentarán en cajas especiales de texto como esta:

```
import math
e=0.3
M=0.5
E=M
Eo=2*M
while abs(E-Eo)>0.01:
    Eo=E
    E=M+e*math.sin(E)
print("E = ",E)
```

E = 0.6886561865220447

¿Puede el lector adivinar qué hace este algoritmo? Si no lo hace, espero que sepa en qué lenguaje de programación está escrito.

²<https://jupyter.org>

³<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

1.4. Mecánica celeste en Python

Es casi imposible escribir un libro con algoritmos sin comprometerse con un lenguaje de programación específico (especialmente si queremos que los algoritmos funcionen.) En el caso de esta edición del libro, el lenguaje elegido es Python.

Esta siempre será una apuesta arriesgada. Aunque la mecánica celeste y sus algoritmos no pasarán de “moda”, los lenguajes de programación “van y vienen”. Es un hecho (poco reconocido) que cientos de libros científicos acumulan polvo por haber comprometido su contenido con lenguajes de programación que hoy no son tan populares (BASIC o Pascal por ejemplo).

No sabemos si Python y este libro sufrirán a la larga la misma suerte. Pero hay tres hechos que *sugieren* que la popularidad de este lenguaje podría durar más de lo esperado (o al menos esa es mi esperanza).

El primero es que su sintaxis es muy similar a la del “lenguaje natural”. Considere, por ejemplo, el algoritmo presentado antes (que ya lo sabe, está escrito en Python) o el siguiente algoritmo, aún más simple:

```
from math import pi
for n in range(1,5):
    print("pi a la",n,"es",pi**n)
```

```
pi a la 1 es 3.141592653589793
pi a la 2 es 9.869604401089358
pi a la 3 es 31.006276680299816
pi a la 4 es 97.40909103400242
```

Es difícil que estos algoritmos se escriban de manera tan natural en casi cualquier otro lenguaje de programación popular en ciencia (C, FORTRAN o Java) como se pueden escribir en Python. Este hecho, no solo facilita el aprendizaje del lenguaje, sino también la legibilidad de los algoritmos.

El segundo hecho que demuestra el promisorio futuro de Python como lenguaje de la computación científica, es la creciente cantidad paquetes, en todas las disciplinas de la ciencia y la técnica, que se escriben permanentemente en este lenguaje y que están disponibles en [repositorios públicos](#)⁴. Además, herramientas informáticas muy conocidas (bibliotecas de rutinas, bases de datos, sistemas de información, etc.) escritas originalmente en otros lenguajes, han sido ahora traducidas a Python (*pythonizadas* si quieren) con el único propósito de que puedan ser usadas por la creciente comunidad de desarrolladores en este lenguaje.

Python se está convirtiendo, y esta es una conjectura mía, en depositario de décadas de experiencia en ciencia computacional. ¿Cambiará esta tendencia pronto? Lo dudo (o al menos así lo espero, por el bien de este libro).

Una última razón, pero no por ello, menos importante, para elegir Python como el idioma oficial de los algoritmos en este libro es la existencia de una biblioteca gráfica, robusta y bien documentada, escrita para este lenguaje. Me refiero por supuesto a [matplotlib](#)⁵.

Con la excepción de paquetes científicos que incluyen avanzadas facilidades de graficación, tales como Mathematica, Matlab, o IDL (todos ellos sujetos a algún

⁴<https://pypi.org/project/IPy>

⁵<https://matplotlib.org/>

tipo de pago), la mayoría de los lenguajes de programación dependen, a veces, de complejas bibliotecas gráficas o programas de terceros para hacer, hasta los más sencillos gráficos.

En Python, hacer un gráfico elemental, es tan simple como escribir:

(Algoritmo 1.1)

```
from matplotlib.pyplot import plot
plot([1,2,3,4],[1,4,9,16]);
```

ver Figura 1.2

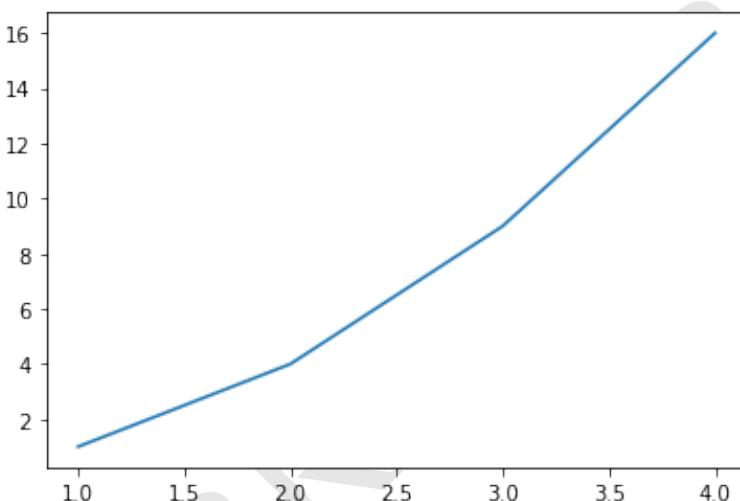


Figura 1.2: Figura correspondiente al código 1.1.

1.5. Mecánica celeste con SPICE

Con el temor de haberlos aburrido ya suficiente con este largo prefacio, no puedo dejar de mencionar aquí, una última herramienta que será protagonista en este libro. Se trata de SPICE, una aplicación desarrollada para la *NASA's Navigation and Ancillary Information Facility (NAIF)*⁶.

SPICE es un sistema de información de uso libre, formado básicamente por una biblioteca de rutinas para realizar cálculos en mecánica celeste y de datos (*kernels*) que permiten, usando esas mismas rutinas, la determinación de la posición y orientación precisa (pasada y futura) de muchos cuerpos del Sistema Solar y de algunos vehículos espaciales lanzados por nuestra especie.

Esta herramienta ha cobrado, en años recientes, una popularidad significativa en la comunidad académica. Sus rutinas y *kernels* están detrás de algunas de los servicios en línea más populares de NASA, tales como el sistema *NASA Horizons*⁷,

⁶<https://naif.jpl.nasa.gov/naif/>

⁷<https://ssd.jpl.nasa.gov/horizons.cgi>

que permite, a través de distintas interfaces, calcular la posición pasada y futura de cuerpos del sistema solar y naves espaciales; o del simulador *NASA's Eyes*⁸ que ofrece vistas en tiempo real de la posición de los cuerpos del sistema solar y de misiones espaciales de la agencia espacial estadounidense.

En este libro usaremos las rutinas y *kernels* de SPICE (a través de la biblioteca *spiceypy*⁹, desarrollada en Python) para ilustrar conceptos, desarrollar ejemplos y resolver problemas que, de otro modo, implicarían un gran esfuerzo algorítmico (el objetivo será no *reinventar la rueda redonda*.)

Al hacerlo, además, el lector, sin importar su nivel, se familiarizará con una herramienta que usan astrónomos e ingenieros aeroespaciales para resolver problemas reales de mecánica celeste. ¡De la teoría a la acción!

De la misma manera como nos preguntamos en el caso de Python, nos formulamos la siguiente pregunta: ¿podría SPICE desaparecer o, mejor, ser reemplazada por un sistema diferente en los próximos años? No podemos asegurarla, pero la cantidad de herramientas que hoy dependen de este sistema de información, hace difícil suponer que podría cambiar radicalmente en el futuro inmediato.

Un último aspecto hace de SPICE una opción muy estable para los propósitos de un libro de texto. La biblioteca de rutinas asociada con el sistema está disponible para un amplio conjunto de lenguajes de programación diferentes a Python. Familiarizarse con las rutinas y *kernels* de SPICE aquí, será suficiente para que pueda usarlo con lenguajes como C/C++, FORTRAN e IDL.

A continuación, y a modo de ilustración, presento un algoritmo, escrito con SPICE, para calcular la distancia de la Tierra al Sol durante el eclipse total de Sol del 29 de mayo de 1919 en el que se obtuvieron las primeras evidencias empíricas de la relatividad general y con el que abrimos este prefacio. Naturalmente, este algoritmo es mucho más complejo (y menos natural) que los que escribí antes, pero ilustra el poder de esta herramienta para obtener resultados interesantes con muy poco esfuerzo computacional.

```
import spiceypy as spy
spy.furnsh('pymcel/data/naif0012.tls')
spy.furnsh('pymcel/data/de430.bsp')
et=spy.str2et("05/29/1919 09:08:00 UTC-3")
sol,tluz=spy.spkgeo(10,et,"J2000",0)
tierra,tluz=spy.spkgeo(399,et,"J2000",0)
distancia=spy.vnrm(tierra-sol)
```

Distancia Tierra-Sol durante el eclipse de 1919: 151649284 km

1.6. ¿Qué hace distinto a este libro?: un decálogo

Para resumir, enumero a continuación las 10 cosas que hacen de este un libro distinto de los muchos que se han escrito en casi 400 años de historia de la mecánica celeste. Este decálogo, como la mayor parte de este prefacio, es, además de una descripción abreviada de las características únicas del libro, una lista de razones que justifican la existencia de un libro más en el “basto océano” de literatura en la

⁸<https://eyes.nasa.gov/>

⁹<https://spiceypy.readthedocs.io/en/master>

materia.

1. ¿Ya les mencione que es un libro para estudiantes de pregrado? Para entender su contenido no es necesario haber visto previamente un curso de mecánica analítica o matemáticas especiales. Solo se necesita una fundamentación mínima en geometría, cálculo y física.
2. El libro ha sido escrito, en la medida de las posibilidades, para ser autocontenido. Todo lo que un lector necesita saber de los fundamentos matemáticos (geometría, cálculo vectorial, ecuaciones diferenciales), los fundamentos físicos (mecánica newtoniana), astronómicos o de computación, ha sido incluído en los capítulos o en apéndices. Esto hace del libro, un texto que puede ser leído o estudiado por personas ajenas a la disciplina, incluso por aficionados.
3. El libro utiliza, como la mayoría de los textos en el área, el *formalismo geométrico y vectorial* de la mecánica para presentar y desarrollar los problemas centrales de la mecánica celeste. Pero también introduce el *formalismo analítico* (mecánica analítica o mecánica clásica) y lo aplica a la mecánica celeste. Es por tanto un libro de mecánica celeste y al mismo tiempo uno de mecánica analítica.
4. El libro no profundiza en todos los temas de la mecánica celeste o la mecánica analítica como lo hacen textos más avanzados. Pero, para un estudiante de pregrado, esta podría ser su primera lectura antes de abordar esos textos.
5. El texto hace un énfasis especial en los algoritmos de la mecánica celeste, que implementa usando códigos en Python, gráficas en matplotlib y, en ocasiones, usando algunas de las rutinas y datos del sistema SPICE de NASA.
6. Todo el libro está disponible como *notebooks* de Jupyter que pueden ser modificados por el lector o ejecutados durante una clase (¡es un libro para enseñar!) Los *notebooks* contienen gráficos interactivos y animaciones que ilustran conceptos que pueden resultar difíciles.
7. El libro no requiere conocimientos previos de programación en Python (aunque tenerlos puede ser muy útil.) En realidad, el libro podría utilizarse como una manera de aprender el lenguaje en contexto, algo que es difícil de conseguir en libros dedicados específicamente a la enseñanza de la programación.
8. Los temas no se desarrollan en el orden en el que aparecieron en la historia: problema de los dos cuerpos → teoría de perturbaciones → problema de los tres cuerpos → mecánica celeste relativística, etc. He preferido presentarlos como me hubiera gustado conocerlos desde el principio, siguiendo un orden más lógico y un poco atemporal. Esta es la manera en la que, creo, un viajero en el tiempo, que retrocediera a 1700, se lo explicaría a un sorprendido Newton.
9. A pesar de lo anterior, la historia es importante en el libro. A través de los capítulos y en recuadros especiales he incluido anécdotas y biografías que permitirán hacerse a una idea del contexto en el que surgieron las principales ideas de la mecánica celeste y la mecánica analítica y de los personajes, hombres y mujeres, que las concibieron.

10. Por muchas de las razones descritas arriba podría decirse que este es un libro “moderno” de mecánica celeste. uno que en lugar de ocuparse de llenar cientos de páginas con sesudos y rigurosos desarrollos matemáticos, le apunta directamente a dar vida a esas ideas y a ofrecer las herramientas prácticas para su aplicación.

Jorge I. Zuluaga

Febrero 19 de 2020

BORRADOR

BORRADOR

Capítulo 2

Agradecimientos

Así como no hay *vacas esféricas en el vacío*, tampoco existen los *autores cilíndricos que escriben aislados*. La elaboración de este libro ha sido determinada y afectada por una multitud de factores y personas a los que no puedo dejar de mencionar.

En primer lugar, quiero agradecer a todos **los estudiantes del pregrado de astronomía** que tomaron el curso de Mecánica Celeste durante los años en los que elaboré las notas que sirvieron de base para este libro. Agradezco su paciencia y sus preguntas en clase que me ayudaron a enriquecer el texto, concentrarme en puntos difíciles y escoger mejor los temas más interesantes. También fue de gran valor los errores que me ayudaron a detectar en las primeras versiones de las *libretas* de Jupyter que son la base del texto. Entre ellos, quiero resaltar a **Andrés Gómez**, quien fue mas lejos aún al revisar críticamente el contenido de algunas *libretas* como lo haría un colega o un editor. Adicionalmente, sus impecables soluciones de los problemas inspiraron una parte del material que he incluido en esta edición del libro.

Una buena parte de la primera versión de las notas del curso fue **transcrita a LaTeX** por el hoy Astrónomo **Bayron Portilla** (en ese entonces mi tallerista del curso). En un momento dado, nos propusimos, incluso, escribir juntos el libro. Sin embargo, nuestras ocupaciones fueron dilatando el proyecto hasta que decidí emprender este proyecto en solitario y partiendo de las *libreta* de Jupyter que laboree posteriormente. Aún así, reconozco y agradezco el esfuerzo que hizo Bayron en esas primeras notas, en las que además exploramos las mejores maneras de organizar los temas del curso. Tal vez en el futuro retome con él algunas de esas notas iniciales con miras a un texto avanzado en la materia donde podamos, por ejemplo, abordar los tópicos que se quedaron por fuera de este libro. En el mismo sentido debo también agradecer al Doctor **Andrés Pérez**, ahora un exitoso astrónomo, quién en sus años como estudiante se ofreció también a transcribir en limpio muchas de mis notas de tablero. El documento resultante que nunca logramos editar apropiadamente todavía lo uso como material de consulta en mis clases. Gracias Andrés por tu dedicación durante esos meses a poner en limpio el sucio de mis tableros.

Estoy también en deuda con **Miguel Vásquez**, el mejor de los talleristas que he

tenido en mi carrera como profesor (ahora es un Astrónomo). Miguel realizó una juiciosa tarea de búsqueda de problemas, transcripción de los mismos al formato de Jupyter y, más importante, preparación en el mismo formato de su solución. Todo, mientras mantenía una estrecha relación con los estudiantes (mucho mejor que la mía como profesor, debo admitir) que le permitió entender sus necesidades, evaluar y ajustar el grado de dificultad de los problemas y recoger correcciones y sugerencias a las notas. **Muchos problemas** incluidos en este libro se basan en el trabajo original de Miguel al que debo hacer un sentido reconocimiento aquí.

Agradezco también a los maestros que me motivaron a estudiar física teórica durante el pregrado y el posgrado, muy a pesar de mi monocromática pasión por la astronomía. Esto me permitió entender, apreciar y abordar mejor los aspectos teóricos de la mecánica celeste. En particular, mis agradecimientos van para los profesores **Lorenzo de la Torre, Alonso Sepúlveda, Jorge Mahecha, William Ponce y Boris Rodríguez**. A través de sus propios manuscritos, conocí (y espero haber aprendido con el ejemplo) el “arte” de escribir libros de texto. El estilo, profundidad y cuidado de sus **notas de clase, libros publicados e inéditos**, han sido imitados sistemáticamente en este libro.

Agradezco a la **Universidad de Antioquia** y en particular a las autoridades del **Instituto de Física** y la **Facultad de Ciencias Exactas y Naturales**, por otorgarme el beneficio de un año sabático, durante el cuál pude, entre otras cosas maravillosas, escribir la primera versión completa de este libro. Mi reconocimiento y agradecimiento además para los **profesores del pregrado de Astronomía**, en especial a mi *parcero* Pablo Cuartas, que recibió mi carga académica y de investigación durante ese año en el que estuve escribiendo.

Finalmente, pero no menos importante, quiero agradecer a mi familia, **Olga y Sofía**. A ellas les toco la peor parte; es decir, soportarme un año entero en la casa, escribiendo en pijamas (o mejor hablando solo, por yo no escribo sino que hablo con el computador) y prestándoles, a veces, menos atención de la que les presto incluso en situaciones normales. Este libro esta dedicado a ellas.

Capítulo 3

Introducción

3.1. ¿Cómo se organiza este libro?

Como mencionamos en la [Sección 1.6](#), una de las cosas hace a este libro diferente de otros textos de mecánica celeste, es la manera y el orden particular en el que se desarrollan los temas. El libro esta dividido en tres grandes partes:

- Los fundamentos matemáticos y físicos.
- Mecánica celeste usando vectores y geometría (formalismo vectorial de la mecánica).
- Mecánica analítica (formalismo lagrangiano y hamiltoniano) y su aplicación en mecánica celeste.

En los siguiente párrafos encontrarán una síntesis *narrada* del libro; algo así como una *tabla de contenido comentada* que le permitirá al lector, no solo orientarse en el texto, sino también entender la manera como se encadenan cada una de sus partes.

Y es que todo libro debería contar una *historia*. En los textos académicos, lamentablemente, esa “vocación” narrativa parece perderse en medio de figuras, teoremas y algoritmos. Esta sección puede ser entonces entendida, como un esfuerzo para esbozar la *historia* que se hila a través de sus capítulos.

- **Parte 1: Fundamentos matemáticos y físicos.** Antes de comenzar, respasaremos algunos temas de matemáticas y de física necesarios para estudiar mecánica celeste. Si bien el lector debería estar familiarizado con la mayoría de estos temas, he decidido incluir este capítulo no solo para hacer al texto autocontenido, sino también con el propósito de compilar resultados útiles, definiciones y algoritmos, en el formato y notación del texto, que se usarán en capítulos posteriores.

- **Capítulo 4.** Algunos consideran a la mecánica celeste un área de las matemáticas aplicadas. En ella confluyen técnicas matemáticas de todos los orígenes. Por esta misma razón para comprender incluso los aspectos

más básicos de la teoría es necesario contar con una sólida fundamentación matemática. Por razones de espacio no podemos cubrir todos los temas relevantes en esta sección, pero nos hemos concentrado en dos de particular importancia en todo el texto:

- ???: El cálculo infinitesimal fue *descubierto* por Isaac Newton a finales de los 1600 (y más tarde descubierto independientemente también por Gottfried Leibniz), inspirado, en parte en problemas mecánicos. Estos métodos matemáticos permitieron a Newton, sus contemporáneos y suscesores resolver los complicados problemas de la mecánica celeste que inauguraron la disciplina. Por la misma razón es indispensable que el lector repase las cantidades y resultados centrales de este método analítico, que es justamente el tema de esta sección. Al hacerlo aprovecharemos además para recoger algunas definiciones y resultados importantes de la geometría y el cálculo de vectores, los elementos básicos de la teoría de ecuaciones diferenciales y del más *exótico* cálculo de variaciones. Ninguno de los apartes de este capítulo cumple funciones *decorativas* o es completamente prescindible. A pesar de parecer una sección ajeno al libro, un material que debería dejarse solo a los autores expertos en el tema, en realidad todos los resultados expuestos aquí serán usados en el resto de capítulos.
- ???: En esta sección nos concentraremos en repasar (o presentar) las propiedades de las figuras cónicas, su definición y descripción geométrica más general, así como su descripción algebraica. Las cónicas juegan un papel central en la mecánica celeste y estar familiarizado con ellas, permitirá resolver más fácilmente problemas físicos relativamente complejos. Estudiaremos esta familia particular de curvas, tanto en el plano, como en el espacio de tres dimensiones. Con este propósito, introduciremos aquí el tema de las rotaciones en dos y tres dimensiones (ángulos de Euler) que son usados con frecuencia en la mecánica celeste pero también en la mecánica analítica.
- ???. Es casi imposible presentar la mecánica celeste y menos aún la mecánica analítica, sin repasar primero las definiciones, postulados y proposiciones de la mecánica básica, o mecánica newtoniana, como se la llama comúnmente. Este capítulo está justamente dedicado a presentar el que llamaremos **formalismo vectorial o geométrico** de la mecánica, desarrollado a partir de las ideas mismas de Newton pero enriquecidas significativamente por sus sucesores en los siguientes dos siglos. Si bien, de nuevo, este podría parecer un tema *elemental* para tratar en otro libro, la manera en la que se presenta aquí es particularmente única. He tratado de formular las ideas de siempre en un orden más moderno y en algunos casos poco ortodoxo. No pretendo con ello producir *ninguna revolución*, pero al hacerlo, la presentación de los temas centrales del libro se hace más natural. El capítulo se concentra en la mecánica de partículas y sistemas de partículas, sin ocuparse de otros temas interesantes de la mecánica, la dinámica de cuerpos rígidos o de fluídos, que no serán

aplicados en el resto del texto.

- **Parte 2: El formalismo vectorial de la mecánica celeste.** Como veremos a lo largo del libro, la mecánica puede ser presentada usando dos enfoques matemáticas o *formalismos* diferentes. En esta parte del curso nos concentraremos en la formulación geométrica o vectorial de la mecánica celeste, la más popular y la que uso originalmente Newton en sus *Principia* y que fue desarrollada posteriormente por sus sucesores.

- ???. A diferencia de la mayoría de los textos en mecánica celeste, en este libro comenzamos por abordar y estudiar con algún detalle, el más general de los problemas de esta disciplina: el problema de los N cuerpos. En este problema, el reto consiste en predecir la posición y velocidad de muchos cuerpos que interactúan gravitacionalmente. Si bien el problema de los N cuerpos fue posiblemente el último de los grandes problemas de mecánica celeste en ser formulado y abordado rigurosamente en la historia, su presentación temprana en este libro, permitirá introducir resultados y métodos que serán de utilidad para el resto del texto. De particular interés será la introducción en este capítulo de los algoritmos para resolver numéricamente el problema. Estos algoritmos y algunas herramientas computacionales relacionadas, serán muy importante en el resto del texto, para comparar y validar resultados de modelos analíticos. Se presentará también aquí el concepto de integrales de movimiento o *cuadraturas*, uno de los métodos usados clásicamente para extraer información sobre un sistema dinámico sin resolverlo completamente. Este método será usado regularmente en los demás capítulos.
- ???. Una de las idealizaciones más conocidas de la mecánica celeste es aquella que consiste en suponer que cuando dos cuerpos astronómicos interactúan gravitacionalmente, el efecto del resto del Universo es completamente despreciable. Naturalmente, no existe ningún sistema astronómico real que cumpla cabalmente estas condiciones. Todos los sistemas del universo, en sentido estricto, son sistemas de N cuerpos. En este capítulo mostraremos, a través de experimentos numéricos y ejemplos astronómicos reales, que la mayoría de los sistemas astronómicos se pueden analizar dinámicamente como *sistemas de N cuerpos jerárquicos*, es decir, sistemas en los que las partículas se agrupan por pares (pares de partículas, pares de pares, etc.) que se perturban mutuamente. El problema de los dos cuerpos no es, sin embargo, el destino final de la mecánica celeste, sino su punto de partida. Es un resultado útil para estudiar sistemas mucho más complejos. Resolveremos en este capítulo el problema de los dos cuerpos usando el método de las cuadraturas (primeras integrales de movimiento) introducido en el capítulo anterior. Demostraremos que el movimiento relativo de dos cuerpos se realiza sobre una cónica y desarrollaremos en detalle las relaciones entre las propiedades geométricas de esa cónica y las propiedades dinámicas del sistema. Resolveremos también, usando métodos geométricos primero y después métodos del cálculo, el denominado problema de los dos cuerpos en el tiempo, que conducirá a la famosa ecuación de Kepler.

- ???. A pesar del poder que la teoría desarrollada en el capítulo anterior tiene para describir el movimiento de muchos sistemas astronómicos, existen situaciones que escapan a una descripción *kepleriana* del movimiento orbital (incluso, una que incluye perturbaciones). El caso de la Luna, el de algunos cometas perturbados por Júpiter y el de vehículos espaciales modernos, son especialmente significativos. En este capítulo abordaremos, inicialmente, el problema general de los tres cuerpos, es decir, aquel en el que la dinámica no es jerárquica. A diferencia del problema de los dos cuerpos, no se conoce una solución general en términos de funciones analíticas al problema de los tres cuerpos. Una versión restringida de este problema, a saber el *problema circular restringido de tres cuerpos* (CRTBP por su sigla en inglés), tiene propiedades teóricas que han resultado de interés en la descripción de sistemas astronómicos reales. Estudiaremos aquí en detalle el CRTBP, su descripción dinámica y cinemática, tanto en sistemas inerciales como no inerciales. Introduciremos algoritmos para la solución numérica del problema en el sistema rotante. Encontraremos su constante de movimiento, la *constante de Jacobi* y una aproximación astronómica en términos de elementos orbitales, el *parámetro de Tisserand*. Deduciremos las propiedades y visualizaremos las denominadas *regiones de exclusión* y *curvas de cero velocidad* (conceptos interesantes que permiten, si no predecir dónde estarán los cuerpos, al menos, donde no estarán). Finalmente se deducirán las propiedades de los *puntos de equilibrio de Lagrange* y algunas aplicaciones astronómicas y en mecánica orbital del problema.
- **Parte 3: El formalismo analítico de la mecánica.** En esta parte del libro, introduciremos el *formalismo analítico de la mecánica* y su aplicación en la solución de problemas de mecánica celeste. El formalismo analítico tiene una importancia central en la física que trasciende a la mecánica celeste (se usa por ejemplo para estudiar la dinámica de cuerpos rígidos y sistemas oscilantes, el caos en sistemas dinámicos, la mecánica relativista, el electromagnetismo, la teoría de campos clásica y la mecánica cuántica). Si bien pocas aplicaciones del formalismo, distintas a la mecánica celeste, se desarrollara en este texto (como si sucede en algunos textos avanzados de mecánica clásica) los fundamentos teóricos presentados en esta parte le permitiran al lector abordar el estudio de esas otras disciplinas en textos específicos de mecánica analítico o en textos más avanzados.
 - ???. En este capítulo se introducen los principios y teoremas centrales del formalismo analítico de la mecánica, en particular los principios de Alambert-Lagrange y de Hamilton. Haremos aquí, un especial énfasis en las motivaciones teóricas que llevaron a matemáticos y físicos de los 1700 a introducir este formalismo (un tema en el que los textos más avanzados de mecánica clásica, apenas si consideran.) Se introducirá aquí la función lagrangiana, las ecuaciones de Lagrange y, a través de la aplicación del cálculo variacional, se deducirán las ecuaciones generales de Euler-Lagrange (que tienen una aplicación amplia en muchas áreas de la física). Como un elemento novedoso se presentarán en este capítulo algunos algoritmos aplicados al formalismo lagrangiano, y en

particular a la comprensión mejor del principio de Hamilton y los métodos del cálculo variacional. Con los elementos básicos del formalismo lagrangiano a la mano, procederemos a aplicarlo en la solución de problemas concretos en mecánica celeste. Para ello presentaremos, primero, resultados importantes sobre la relación entre las simetrías de la función lagrangiana y las cantidades conservadas en el movimiento (teorema de Noether). A partir de allí, procederemos de forma similar a como lo hicimos con el formalismo vectorial, a resolver el problema general de los N cuerpos y el de los dos cuerpos. Deduciremos el lagrangiano de los N cuerpos y de sus simetrías obtendremos las cantidades conservadas en el sistema. Pero ¿de qué sirve deducir los mismos resultados que ya habíamos visto en el capítulo correspondiente de la segunda parte? Usaremos lo que sabemos de mecánica celeste para ilustrar el poder del formalismo lagrangiano frente al formalismo vectorial. Posteriormente, abordaremos el problema de los dos cuerpos usando el formalismo lagrangiano. En este caso, a diferencia del problema de los N cuerpos, tendremos una novedad. En lugar de restringirnos al caso de la gravitación Newtoniano, estudiaremos aquí el problema más general de sistemas de dos cuerpos sometidos a fuerzas centrales con un potencial generalizado. Los resultados obtenidos aquí, tendrán un rango más amplio de aplicación. Podrán por ejemplo usarse para estudiar la física de sólidos, moléculas y átomos, pero también la mecánica celeste postnewtoniana. Estudiaremos, en este contexto, el problema de fuerzas centrales reducido a una dimensión, el potencial efectivo (y las correspondientes zonas de exclusión). Para el caso del potencial newtoniano deduciremos la denominada ecuación de la forma orbital y resolveremos el problema de los dos cuerpos a partir de ella. Para el caso de un potencial general, pero no muy distinto del potencial Newtoniano, estudiaremos el denominado *avance del perihelio* uno de los resultados teóricos de la mecánica celeste que a la larga serían de la mayor importancia histórica al ofrecer las primeras evidencias de la validez de la teoría general de la relatividad.

- ???. En este capítulo abordamos el más general (y poderoso) formalismo analítico de la mecánica: el formalismo Hamiltoniano. Después de discutir las motivaciones para la introducción de este formalismo (motivaciones de naturaleza principalmente geométrica), deduciremos de forma heurística las ecuaciones canónicas (de primer orden) de Hamilton; introduciremos la función Hamiltoniana y demostraremos su equivalencia con las ecuaciones (de segundo orden) de Euler-Lagrange. Ilustraremos el poder del formalismo y la descripción de los sistemas en el denominado *espacio de fase*; para ello nos valdremos inicialmente de sistemas dinámicos simples (péndulos y bloques), como hicimos en el primer capítulo de esta parte. Posteriormente abordaremos (sin el detalle en el que lo hicimos en el caso del formalismo Lagrangiano y por las obvias analogías entre los dos formalismos) el tema de las simetrías y las cantidades conservadas, e introduciremos los útiles *corchetes de Poisson*, como herramienta matemática para estudiar dichas simetrías. Escribiremos los hamiltonianos del problema general de los N cuerpos, el del

problema de los dos cuerpos y el del problema circular restringido de los tres cuerpos, y redescubriremos, usando los elementos de este nuevo formalismo, las propiedades ya conocidas de estos sistemas. Una de las aplicaciones más poderosas del formalismo Hamiltoniano, se consigue al aprovechar las simetrías de los sistemas gravitacionales, para, a través de transformaciones de *coordenadas* en el espacio de fase, escribir formas simplificadas de los Hamiltonianos. Estas formas simplificadas, además, permiten aplicar de forma más directa la teoría de perturbaciones y así estudiar sistemas muy complejos (un tema que no está incluido en este libro.) En este capítulo introduciremos, primero, el tema de las transformaciones canónicas, que son transformaciones de coordenadas en el espacio de fase que mantienen la *estructura hamiltoniana* de los sistemas (es decir, que hacen que los sistemas sigan siendo descritos con las ecuaciones canónicas). Nos concentraremos, especialmente en el formalismo de la función generatriz de las transformaciones canónicas. A continuación, aplicando la teoría de transformaciones canónicas presentaremos el método de Hamilton-Jacobi que permite, entre otras cosas, encontrar sistemas de coordenadas que simplifican significativamente la descripción de ciertos sistemas físicos. En particular utilizaremos este formalismo para deducir, en el problema de los dos cuerpos, el Hamiltoniano del sistema en términos de elementos orbitales; en particular, en términos de funciones específicas de esos elementos orbitales, que hacen lo más simple posible el hamiltoniano del sistema. El resultado más importante de este capítulo será la deducción de las denominadas *variables de Dalaunay* que son de gran utilidad y poder en la mecánica celeste moderna y posiblemente el punto de partida de algunos textos de mecánica celeste avanzados.

Todos los capítulos hasta aquí contarán con un conjunto completo de preguntas, ejercicios y problemas, que permitirán al lector poner a prueba los conocimientos adquiridos y las habilidades desarrolladas, pero también, descubrir como estas ideas, métodos y herramientas, se aplican en otras situaciones específicas.

3.2. ¿Cómo usar este libro?

Este libro puede ser utilizado de tres formas diferentes:

1. Como un texto para el *autoaprendizaje* de la mecánica celeste y la mecánica analítica. Estudiantes y profesionales de muchas disciplinas, se pueden valer de él para tener su primer acercamiento a estas disciplinas.
2. Como el texto guía de un primer curso de mecánica celeste y de mecánica analítica. El texto es una fuente de lecciones y problemas útiles para organizar un curso de pregrado.
3. Como material de referencia para estudiantes y profesionales. Muchas fórmulas, algoritmos, e incluso anécdotas e historias interesantes, podrían resultar útiles para quiénes ya tienen una formación en el área.

Como **texto para el autoaprendizaje**, recomiendo **leer el texto en su totalidad** incluyendo la primera parte de Fundamentación matemática y física, en la que se encuentran algunos elementos teóricos requeridos para el resto del libro.

Para quiénes tengan una formación avanzada en física, astronomía o ingeniería, es posible que una buena fracción de los temas de esa primera parte resulten sencillos y puedan obviarse. Sin embargo, aunque los tópicos tratados allí parezcan conocidos (al menos nominalmente), su tratamiento puede resultar novedoso. Si este es su caso, no deje de echarle una mirada a esos primeros capítulos. En particular recomiendo revisar, como mínimo, las secciones dedicadas a la solución numérica de ecuaciones diferenciales, incluyendo las ecuaciones de movimiento en mecánica newtoniana, los rudimentos de cálculo variacional y la dinámica en sistemas rotantes, donde podrían encontrarse las diferencias más significativas respecto a los textos canónicos de matemáticas y física, y donde además se introducen herramientas algorítmicas que serán de uso muy corriente en el resto del libro.

El uso ideal de este libro es como **texto guía** de un primer curso de mecánica celeste y mecánica analítica. El libro fue escrito a partir de la experiencia de más de 5 años ofreciendo el curso en el pregrado de astronomía de la Universidad de Antioquia (en Medellín, Colombia). Por la misma razón, la extensión y organización particular del texto, se adapta de forma *precisa* a las condiciones propias de un curso universitario de un semestre de duración (cuatro meses efectivos de lecciones.) El curso se ha ofrecido exitosamente a estudiantes que han aprobado los cursos básicos de física (hasta el tema de oscilaciones y ondas) y de cálculo (incluyendo cálculo vectorial y ecuaciones diferenciales.)

Todos los capítulos del libro han sido dictados dentro del plazo del curso. Sin embargo, dependiendo del nivel académico de los estudiantes y de su independencia intelectual, el curso puede dictarse sin incluir todos los temas de la primera parte.

Por mi experiencia dictando el curso, el repaso de los fundamentos puede resultar extenso (como mínimo toma un mes que es justamente el período en el que los estudiantes tienen una motivación y disposición mayor, además de menos distracciones de otros cursos.) Sugeriría, entonces, que de sacrificarse algunos temas de esa parte, se asigne la lectura independiente a los estudiantes de los temas mejor conocidos y se evalúe a través de la lista de problemas incluidos al final de los capítulos de esa parte.

Como mencioné en la [Sección 1.3](#), y se detallará abajo, el libro fue escrito usando *libretas de Jupyter*, una por cada clase (a lo sumo se pueden dictar dos clases con cada libreta). Es decir, el número de *libretas* y su organización puede ofrecer una idea del programa detallado de actividades del curso o del plan de lecturas.

3.3. Mecánica celeste en *libretas*

El libro ha sido concebido, escrito y compilado enteramente usando *libretas* de *Jupyter*. Las libretas, que están disponibles en la versión electrónica del texto, son archivos en un formato especial (no son programa de Python, ni páginas web) que pueden ser visualizadas y ejecutadas usando un navegador de Internet.

El uso de las libretas no es indispensable para entender el contenido del libro, pero puede ofrecer una experiencia interactiva muy enriquecedora y a veces acelerar el proceso de aprendizaje. El uso de las libretas en clase puede, además, hacer

más dinámica y amena la interacción entre el profesor y los estudiantes.

Para hacer uso de las libretas se debe contar con un **computador de escritorio** que use cualquier sistema operativo (Windows, Linux o MacOS). Por la misma razón, en caso de usarla, recomiendo que el curso se desarrolle en una sala de computo. Para ejecutar las libretas es necesario instalar primero el interprete y la biblioteca base del lenguaje Python, un conjunto específico de paquetes y el sistema Jupyter, además de varias de sus extensiones (los detalles se presentan en la siguiente sección.)

La **versión en línea**¹⁰ de este libro (páginas web), puede ser también una alternativa a las libretas de Jupyter. Este formato tiene la ventaja que solo requiere un dispositivo con conexión a Internet (de escritorio o móvil) y puede manipularse en cualquier contexto. Aunque la versión web carece de casi todas las características interactivas de las libretas de Jupyter, en ella encontrarán, además de todos los algoritmos y gráficos, animaciones y otros elementos de *hipertexto*.

3.3.1. Instalación de las libretas

Para aquellos que deseen aprovechar las libretas de Jupyter como medio didáctico, se ofrece a continuación una guía básica de cómo preparar un computador para ejecutarlas. Instrucciones adicionales pueden encontrarse en la versión en línea del libro.

1. **Instalación del lenguaje Python y las bibliotecas básicas del lenguaje.** El primer requisito para utilizar las libretas es instalar el interprete y las bibliotecas del sistema del lenguaje Python. Existen diversas maneras para hacerlo en cada sistema operativo y abundantes instrucciones en Internet. Mi recomendación es utilizar el sistema [Anaconda](#)¹¹ que ofrece, en una plataforma integrada, los archivos del lenguaje Python, una amplia diversidad de paquetes científicos, el sistema Jupyter y todas las herramientas necesarias para la instalación de otros paquetes.
2. **Descargar las libretas.** Una vez haya instalado Python y Jupyter, puede descargar las libretas del libro los archivos adicionales requeridos por ellas del sitio web del libro. Para ello siga las instrucciones provistas allí.
3. **Ejecución de pruebas.** Para verificar si las libretas funcionan correctamente, una vez descargadas, busque y abra la libreta Pruebas.ipynb. Una vez abierta ejecute todas sus celdas (Cell / Run all). Si la ejecución se realiza completa, en la última celda aparecerá un reporte completo con los resultados de la prueba. Si alguna de las prueba individuales falla, es posible que sea necesario instalar paquetes, datos adicionales y otras dependencias.
4. **Instalación de dependencias.** Para instalar todas las dependencias del libro abra la libreta Instalacion.ipynb y siga las instrucciones descritas allí.

¹⁰<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

¹¹<https://www.anaconda.com>

3.4. Idioma y Notación

3.4.1. Palabras extranjeras y guía de pronunciación

El libro está escrito en español. Sin embargo, y como sucede con todas las ciencias, habrán muchos apartes en los que es necesario introducir términos técnicos y acrónimos procedentes de la lengua inglesa. En estos casos las palabras y acrónimos se presentarán en itálica. Así por ejemplo, al referirnos al problema matemático de resolver la ecuación de movimiento de una partícula hablaremos del *initial value problem* o su acrónimo *IVP*, en contraposición al *boundary condition problem*. Por otro lado en el ?? estudiaremos el *CRTBP* o *circular restricted three body problem*.

Muchos de los científicos (hombres y mujeres) que han contribuído con el desarrollo de la mecánica celeste en sus cuatro siglos de historia, tienen nombres y apellidos no hispanos. Su correcta pronunciación, especialmente en el caso de autores franceses o de origen germano, es difícil para quienes no hablamos las lenguas de esos pueblos.

Un caso notable, por ejemplo, es el nombre de la matemática alemana *Emmy Noether*. En castellano la mayoría pronunciaríamos “emi noeter” o “emmi neder” (siguiendo la tradición inglesa con la que estamos más familiarizados.) Como una primera guía para la correcta pronunciación de estos nombres, a lo largo del libro presentaremos “transliteraciones” al castellano, indicando, entre comillas las letras y palabras más cercanas que un hispanohablante podría usar. Así por ejemplo “niuton” será la transliteración fonética de Newton y la pronunciación “correcta” (en alemán) del nombre de Emmy Noether, será “emmi noutar”.

Para hacernos a una idea fonética más precisa nos apoyaremos a lo largo del libro de la increíble colección compilada en [este sitio web¹²](#) que ofrece pronunciaciōnes en línea, en decenas de idiomas, de miles de nombres, palabras y frases. Allí encontrará por ejemplo la pronunciación correcta, en su idioma original del nombre [Emmy Noether¹³](#).

¿Es todo esto indispensable para entender la mecánica celeste o la mecánica analítica?. Ciertamente no. Pero no solo de teoría vivimos los humanos. La comunicación y socialización es central al proyecto científico y es bueno entender y hacerse entender especialmente en contextos internacionales.

3.4.2. Siglos y décadas

La historia de la mecánica celeste y analítica, así como la historia de las áreas de la física y las matemáticas con las que se relaciona, es fascinante. En el libro, como detallamos en la próxima sección, incluiremos abundantes referencias históricas sobre los personajes y los momentos claves en el desarrollo de las ideas de la mecánica celeste.

Para referirnos a los siglos, sin embargo, nos desviaremos de las reglas convencionales del español. Según esas reglas al período comprendido, por ejemplo, entre 1701 y 1800, se lo llama el siglo XVIII. Para este autor, la notación usando números romanos, si bien ampliamente aceptada, es confusa y exige realizar operaciones mentales innecesarias (número romano → número indoarabigo → restar uno →

¹²<http://forvo.com>

¹³<https://es.forvo.com/search/Emmy%20Noether/de/>

multiplicar por 100).

En los sucesivo para referirnos al período comprendido entre 1700 y 1799 (comenzando en el año cero y no en el año uno como dicta la regla) hablaremos de **los 1700**. Así mismo el siglo XX será **los 1900** y así sucesivamente.

Dado que en las reglas establecidas del español, los 1900 hacen referencia en realidad a la década entre 1901 y 1910, cuando queramos referirnos a un período de diez años siempre usaremos explícitamente la palabra **década**: década de 1680, década de 1960, etc.

No pretendó, con este acto de rebeldía *idiomática*, cambiar el castellano. Pero sí, al menos en lo que respecta a este libro, facilitar la lectura de los períodos históricos.

3.4.3. Notación matemática

Todos los libros de ciencias físicas o matemáticas se “casan” con una notación específica. La elección de la notación, no es sin embargo una tarea sencilla, en tanto son muy comunes los casos de textos que en virtud de su notación se hacen prácticamente ilegibles aunque traten los mismos temas o problemas de otros que usan notaciones más comunes.

Pensando justamente en esto, he tomado la decisión de utilizar, en la medida de las posibilidades, la misma notación de algunos textos clásicos de mecánica celeste, que se diferencia, a veces significativamente, de la que utilizan libros de matemáticas e incluso de física, con los que el lector puede estar familiarizado.

El lector encontrará los detalles específicos de la notación usada en el libro en la [Sección 4.1](#).

3.5. Elementos no textuales

Para facilitar la lectura del libro y hacer de la experiencia de leerlo algo más agradable e incluso excitante, el texto contiene una serie de elementos gráficos con los que debemos familiarizarnos.

3.5.1. Cajas de texto

Mucha información importante texto se presenta en *cajas* independientes al texto principal y cuyas características gráficas resaltan del resto del documento. En particular existen 5 tipos de cajas:

- **Resumen del capítulo.** Esta caja aparece normalmente al principio de cada capítulo y contiene una breve síntesis del mismo. No deje de leer este resumen para identificar los temas centrales de cada parte del libro. El profesor podría usar la información contenida allí para definir los objetivos específicos de la evaluación.
- **Notas.** A veces es necesario desviarse un momento del hilo del texto para aclarar o ampliar asuntos relacionados con la notación, los paquetes y algoritmos utilizados, o simplemente llamar la atención sobre un asunto importante. A continuación se muestra un ejemplo de una *caja de nota*.



Nota

El lenguaje Markdown. La mayor parte del contenido textual de este libro, ha sido escrito en las celdas de libretas de Jupyter en un lenguaje de descripción de documentos conocido como *Markdown*. Puede explorar la sintaxis del lenguaje, o bien desplegando el contenido de las *celdas* de las libretas, o bien consultando la abundante [documentación en línea^a](#).

^a<https://markdown.es/>

- **Definiciones.** Muchas cantidades físicas y algunos conceptos claves requieren una definición rigurosa. Este es el rol justamente que juegan las *cajas de definición*. A diferencia de las cajas de Resumen y Notas, las cajas de *Definición* están numeradas (como las figuras o las ecuaciones), de modo que sea más fácil referirse a ellas.

Definición 3.1

Mecánica celeste. Llamamos *Mecánica Celeste* a la disciplina científica que aplica las leyes de la mecánica para estudiar el movimiento de cuerpos bajo la acción dominante de la gravedad. Dado que solo en lugares lejanos a la superficie terrestre (normalmente fuera de su atmósfera), la gravedad es la fuerza dominante, la mecánica celeste normalmente describe el movimiento de cuerpos astronómicos (desde partículas pequeñas, hielo o polvo interestelar, hasta planetas y estrellas) y de vehículos espaciales. En este último caso se habla normalmente de *Mecánica orbital*.

- **Teoremas, postulados y leyes.** Como las definiciones, en muchas ocasiones será indispensable separarnos un momento de una explicación para formular más rigurosamente un resultado, normalmente obtenido por razonamiento deductivo en el marco de una teoría (teoremas, lemas, colorarios) o por razonamiento inductivo a partir de la experiencia (leyes y postulados). Para hacerlo usaremos cajas de texto con una numeración independiente de aquella usada para las definiciones. Sin embargo, es importante aclarar que en el caso de los denominados *teoremas* me he abstenido de usar sistemáticamente esta palabra en el encabezado de los respectivos recuadros. En su lugar he decidido imitar a algunos autores clásicos (en particular a Euclides) que usaban sistemáticamente la palabra **proposición** en lugar de teorema para referirse a afirmaciones demostrables. Es decir, en este libro, una *proposición* será un resultado importante que puede estar o no demostrado en el texto. Al hacerlo quiero evitar posar aquí de matemático, una profesión a la que respeto profundamente¹⁴. Aún así, cuando una *proposición* dada corresponda a un teorema bien conocido, usaré la palabra teorema en el título interno

¹⁴Decía el matemático húngaro Paul Eördos (“pol érdos”) que un *matemático es una máquina para convertir café en teoremas*, una frase que aunque parece simplificar la naturaleza de los matemáticos, en realidad demuestra la importancia que tienen los teoremas para esta milenaria profesión.

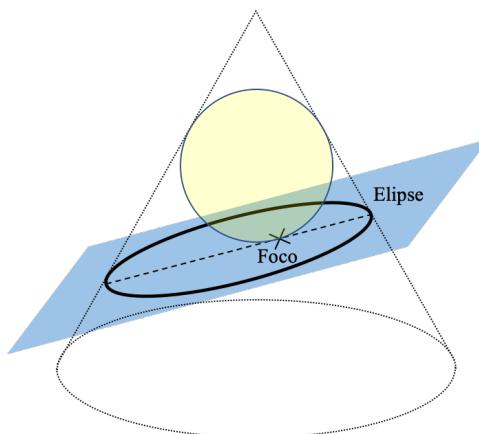


Figura 3.1: Ilustración esquemática del teorema de Danelin.

de la caja. Las dos *proposiciones* mostradas en las cajas a continuación ilustran estos conceptos.

Proposición 3.1

Sistemas de referencia inerciales. Si un sistema de referencia O' se mueve con velocidad constante con respecto a un sistema de referencia inercial O , entonces O' es también un sistema de referencia inercial.

Proposición 3.2

Teorema de Danelin. Dada una esfera tangente a un cono y un plano que corta el cono en un determinado ángulo, el punto de tangencia de la esfera con el plano es uno de los focos de la cónica correspondiente.

- **Un poco de historia.** Finalmente, pero no menos importante, están las anécdotas e historias que contaremos a lo largo de todo el libro. Como se mencionó en el prefacio, la mecánica celeste tiene ya más de 400 años (aproximadamente 100 años más que la mecánica analítica) y cientos de libros y miles de artículos se han escrito en el tema. Es casi imposible hablar de mecánica celeste y analítica, sin mencionar de vez en cuando las historias que rodearon la invención de una técnica, la biografía de alguno de los grandes hombres y mujeres que concibieron las ideas contenidas en el libro o simplemente una anécdota curiosa relacionada con algún tema de interés.

3.5.2. Algoritmos

Como he insistido hasta aquí, una de las novedades más importantes de este libro es el énfasis que he querido dar a los *algoritmos*. Por algoritmo entenderemos

Un poco de historia

¿Kepler o Newton? En el Prefacio daba a entender que la mecánica celeste posiblemente había comenzado con los trabajos pioneros de Johannes Kepler (ver [Figura 3.2](#)). Otros autores van más lejos y apuntan a los astrónomos de la antigüedad y la edad media, especialmente indios, chinos, árabes y griegos, que desarrollaron modelos complejos para la descripción del movimiento de los cuerpos celestes. Los más conservadores apuntan a Sir Isaac Newton, quien después de la publicación de su obra cumbre, los *Principia*, sentó las bases físicas, no solo para la mecánica celeste, sino también, en general, para toda la mecánica.

La razón en este libro para escoger a Kepler, como el *padre* de la disciplina (y en general de la astronomía física) fueron sus contribuciones decisivas y bastante bien conocidas para esclarecer definitivamente la *cinemática* del movimiento planetario. En particular, el descubrimiento (o el enunciado matemático) de sus conocidas *leyes del movimiento planetario* representaron un cambio cualitativo en el desarrollo de la teoría del movimiento planetario e inspiraron en últimas el trabajo de Newton y sus contemporáneos.

Adicionalmente, y esto es aún más importante, Kepler fue uno de los primeros astrónomos modernos (renacentistas europeos) en hacer consideraciones teóricas sobre la causa del movimiento planetario, más allá de ocuparse de su descripción, como lo hicieron la mayoría de los astrónomos de la antigüedad y la edad media. Esto pone a Kepler, entre esos astrónomos, como el primer *astrofísico* de la historia.



Figura 3.2: Retrato de Johannes Kepler, copia de un original de 1610 de pintor desconocido y que se conserva en el monasterio Benedictino de Kremsmünster (Alemania).

aquí pequeños (o no tan pequeños) fragmentos de código (*code snippet* en inglés) que realizan tareas numéricas específicas o son parte de un algoritmo mayor.

He evitado hablar de *programas* o *códigos* para resaltar el hecho de que lo importante en ellos es la lógica de las operaciones y no el lenguaje específico en el que están escritos. A pesar de este esfuerzo por mantener el tema lo más general posible, es virtualmente imposible escribir algoritmos que se puedan ejecutar realmente en las libretas, sin recurrir a ciertas particularidades del lenguaje en el que están descritos, Python.

Existen en general tres tipos de *algoritmos* que encontraremos a lo largo del texto. En primer lugar están los algoritmos más sencillos, aquellos que ejecutan tareas básicas de preparación de datos para algoritmos más complejos. Este es un caso de ellos:

```
a=1
b=-1
c=2
disc=b**2-4*a*c
```

Discriminante = -7.0

Muchos de estos algoritmos simples vienen seguidos del resultado más importante de las operaciones que codifican. En el caso anterior se muestra por ejemplo el valor del discriminante (el valor de la variable disc). El algoritmo (o código) para producir ese resultado:

```
print(f"Discriminante = {disc:.1f}")
```

Pero este algoritmo (y la celda correspondiente) no se muestra en el libro impreso para evitar la proliferación de código irrelevante.

Los algoritmos más complejos pueden, como las ecuaciones, estar numerados:

(Algoritmo 3.1)

```
def calcula_discriminante(a,b,c):
    disc=b**2-4*a*c
    return disc
```

En este caso, el algoritmo contiene una rutina o función, que podría ser usada más adelante, incluso en un capítulo posterior. Todas las rutinas como estas, hacen parte de un paquete incluído con las libretas llamado `pymcel`. Para usar la rutina en el Alg. (3.1) en otra parte del libro se usa:

```
from pymcel.export import calcula_discriminante
d=calcula_discriminante(1,2,3)
```

Cualquier lenguaje de programación moderno depende de numerosas bibliotecas en las que están codificados procedimientos de uso regular o muy especializados. En todos los algoritmos presentados en el libro, siempre que se use una rutina de una biblioteca externa, se presentará el código que hace referencia a la biblioteca de forma explícita. Consider por ejemplo este algoritmo:

(Algoritmo 3.2)

```
#Coeficientes de un polinomio de segundo grado
a=1
b=3
c=-2

#Calcula discriminante
from pymcel.export import calcula_discriminante
d=calcula_discriminante(a,b,c)

#Calcula raices
if d>=0:
    from numpy import sqrt
    x1=(-b+sqrt(d))/(2*a)
    x2=(-b-sqrt(d))/(2*a)
else:
    print("El polinomio no tiene raices reales")
```

El polinomio no tiene raices reales

En él hemos usado la rutina `sqrt` (raíz cuadrada) de la biblioteca NumPy para calcular, en este caso, las raices de un polinomio de segundo grado. Para ello, antes de la línea que usa la raíz cuadrada hemos incluido la instrucción:

```
from numpy import sqrt
```

Aunque en los programas regulares, estas instrucciones se ponen al principio, he decidido colocarlas lo más cerca posible al lugar donde se usan de modo que los fragmentos de código funcionen fuera del contexto del libro. El lector poco familiarizado con el lenguaje Python puede hacer caso omiso a estas instrucciones, que nada le agregan a la lógica de los algoritmos.

Nota

las instrucciones `import` y la velocidad de los programas. Es importante advertir que en algunos algoritmos, usar muchas instrucciones del tiempo `import` entre las líneas de código puede disminuir la velocidad del código. La recomendación general es la de poner este tipo de instrucciones al principio del programa. Así el Alg. (3.2) debería escribirse así:

```
from pymcel.export import calcula_discriminante
from numpy import sqrt

#Coeficientes de un polinomio de segundo grado
a=1
b=3
c=-2
```



Nota (Cont.)

```
#Calcula discriminante
d=calcula_discriminante(a,b,c)

#Calcula raices
if d>=0:
    x1=(-b+sqrt(d))/(2*a)
    x2=(-b-sqrt(d))/(2*a)
else:
    print("El polinomio no tiene raices reales")
```

Otro tipo de algoritmos frecuentes son aquellos que dan como resultado figuras o gráficos. Estos están entre los más interesantes y útiles, aunque pueden ser complicados y causar algo de estupor para los menos familiarizados con el lenguaje de programación. Les recomiendo a todos poner especial atención en estos algoritmos, tratar de entenderlos e imitarlos. Una buena parte de la ciencia que hacemos hoy día depende de producir bonitos productos gráficos que ilustren de forma compacta conceptos o resultados difíciles de describir de otra manera.

Todos los códigos que producen figuras están numerados. Así mismo los gráficos que producen aparecen en el texto, incluso en el impreso, como figuras independientes y numeradas. Por razones de eficiencia en el uso del espacio, algunas de esos gráficos pueden estar en lugares lejanos de la posición del código. Es por esto que en todos los algoritmos que producen gráficos encontraran (en la parte inferior) una referencia a la figura correspondiente.

(Algoritmo 3.3)

```
from numpy import linspace,sin,pi
t=linspace(0,2*pi)
x=sin(t)

import matplotlib.pyplot as plt
plt.figure()
plt.plot(t,x,'k-');

plt.xlabel("t");
plt.ylabel("x(t)");
```

ver Figura 3.3

La mayoría de las figuras del libro han sido elaboradas usando software de diseño independientes. Sin embargo, algunas figuras, especialmente gráficos de datos o resultados de simulaciones, son generadas por las libretas con las que fue escrito el libro. Si bien los algoritmos con los que son creados esas figuras (que llamaremos *gráficos generados*) no aparecen en la versión impresa o en la versión web porque pueden ser muy elaborados e irrelevantes para los fines del texto, si pueden aparecer en las libretas de clase.

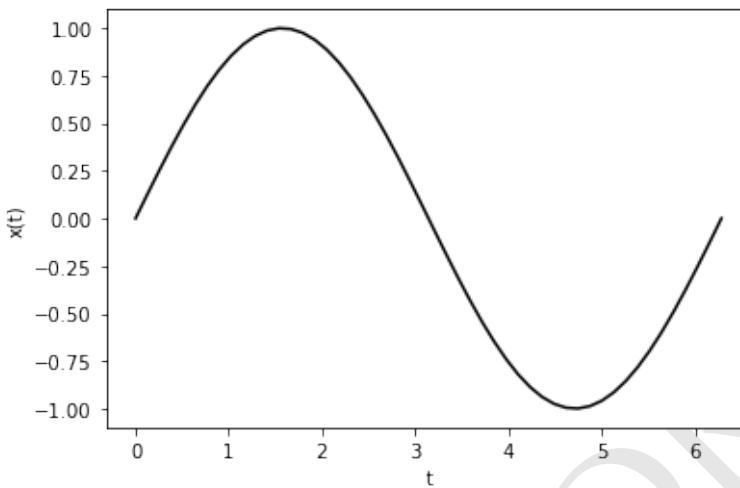


Figura 3.3: Figura correspondiente al código 3.3.

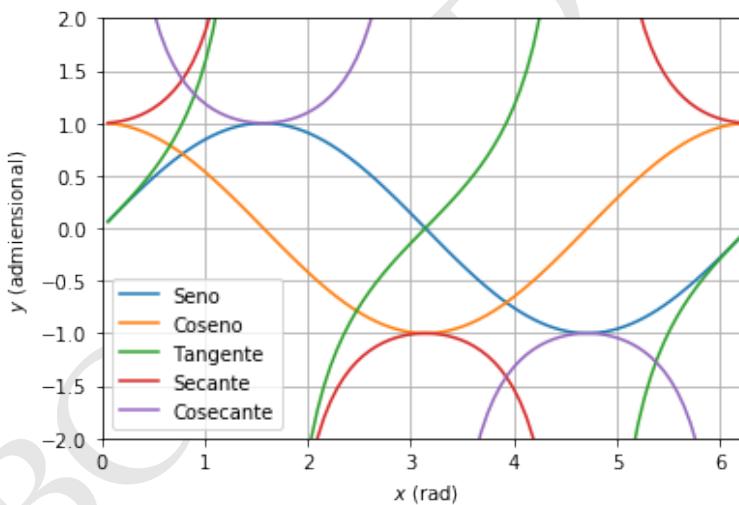


Figura 3.4: Gráfico de las funciones trigonométricas básicas, en el intervalo de interés (gráfico generado).

3.6. Figuras interactivas y animaciones

Uno de las cosas que hace poderosas a las libretas de Jupyter como medios para compartir información o estudiar un tema, es la posibilidad de interactuar directamente con esa información. Esto se consigue modificando el contenido de las celdas de las libretas (código) y ejecutándolas independientemente.

Pero hay otra posibilidad. En muchos apartes del libro se han creado gráficos interactivos y animaciones que permitirán al lector o al estudiante, modificar de forma gráfica (sin ir directamente al código) los parámetros de un algoritmo (gráfi-

cos interactivos) o ver en movimiento figuras que normalmente están estáticas en los libros.

Busque las figuras interactivas en la [versión en línea¹⁵](#) del libro.

BORRADOR

¹⁵<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

Capítulo 4

Fundamentos matemáticos

Resumen

En este capítulo haremos una síntesis práctica de los temas de matemáticas que usaremos en el resto del libro para presentar las teorías y métodos de la mecánica celeste. Repasaremos la geometría de las cónicas (que son la base para describir la trayectorias de cuerpos celestes sometidos a la gravidad newtoniana), en el plano y en el espacio de tres dimensiones. Haremos una síntesis muy práctica de las definiciones y proposiciones de la geometría vectorial, los sistemas de coordenadas, el cálculo diferencial, el cálculo integral y el más exótico pero muy importante cálculo de variaciones. También introduciremos algunos resultados útiles de la teoría de ecuaciones diferenciales y más importante los algoritmos para manipular todas estas cantidades que serán aplicadas a lo largo del texto.

4.1. Vectores y cálculo

En esta sección repasaremos, de manera práctica (y posiblemente poco rigurosa desde el punto de vista matemático), algunos resultados centrales del cálculo infinitesimal y la teoría de ecuaciones diferenciales que serán de utilidad en el resto del libro.

Para quienes conocen bien estos temas, puede servir de motivación para la lectura de esta sección, el hecho de que además de conceptos matemáticos ampliamente conocidos, hemos incluido aquí detalles sobre la **notación matemática, definiciones y teoremas** que usaremos en el resto del libro; escritos todos en un lenguaje muy propio del texto. Tal vez más interesante es el hecho de que a lo largo de esta sección ilustraremos también algunos de los conceptos claves usando **algoritmos**, con lo que sentaremos las bases para todos los desarrollos *computacionales* de los demás capítulos.

Sea que lea esta sección o sea que no lo haga, antes de pasar a los siguientes capítulos intente resolver los problemas al final de este capítulo que están directamente relacionados con los temas de esta sección. Este ejercicio le permitirá valorar

mejor las habilidades matemáticas y algorítmicas que tiene antes de comenzar y que serán indispensable en el resto del libro. Tal vez descubra que después todo no es mala idea hacer este repaso.

4.1.1. Conjunto, tuplas y vectores

Hay tres tipos de entidades matemáticas (además de los números reales y las funciones) que usaremos con frecuencia en este capítulo (y en general en todo el libro):

- **Conjuntos.** Muchas veces nos referiremos aquí a conjuntos (no necesariamente ordenados) de entidades que están relacionadas de alguna manera: las coordenadas de un punto en el espacio de fases, un conjunto de funciones, las ecuaciones diferenciales que describen el movimiento de un sistema dinámico, las partículas que interactúan gravitacionalmente en un sistema, etc. Los elementos de la mayoría de los conjuntos usados en este libro estarán numerados. Así por ejemplo, las masas de un sistema de N partículas, m_0, m_1, \dots, m_{N-1} se representarán como el conjunto:

$$\{m_i\}_{i=0,1,\dots,N-1}$$

Una versión sintética más común de esta notación será $\{m_i\}_N$. En el caso en el que el número de elementos sea claro en el contexto se usará simplemente $\{m_i\}$



Nota

Numeración comenzando en cero. En lo sucesivo numeraremos todas las cantidades físicas y matemáticas (partículas, variables auxiliares, componentes de un vector o una matriz, etc.) comenzando en cero, tal y como se acostumbra en programación. Esta elección facilitará la implementación de las fórmulas en algoritmos y programas de computadora. Si bien la numeración comenzando en cero no es muy común en matemáticas o física, existen justificaciones poderosas para su uso, algunas de las cuales están enumeradas en el documento “[Why numbering should start at zero](#)”^a del maestro de maestros de la programación científica, Edsger Wybe Dijkstra.

^a<https://www.cs.utexas.edu/users/EWD/transcriptions/EWD08xx/EWD831.html>

- **Tuplas.** Las tuplas (pares, tripletas, etc.) son conjuntos ordenados de números reales. Para las tuplas usaremos la notación convencional $(x_0, x_1, \dots, x_{N-1})$, donde los paréntesis, a diferencia de las llaves de los conjuntos más generales, nos permitirán reconocer el hecho de que el orden de los elementos es importante. Las tuplas forman, con el conjunto de los números reales, un *espacio vectorial*. En este espacio se definen las siguientes operaciones básicas:

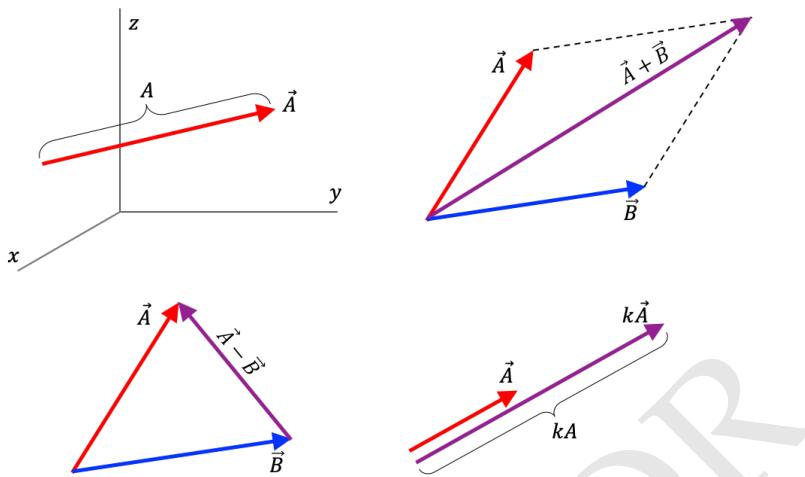


Figura 4.1: Definición geométrica de vector espacial y de sus operaciones básicas (suma, resta y multiplicación por un escalar). Aunque la resta de $\vec{A} - \vec{B}$ es un caso particular de la suma, es importante aquí familiarizarse con la dirección que tiene este vector (va de la cabeza del sustraendo \vec{B} a la del minuendo \vec{A} .)

- Suma:

$$(a_0, a_1, \dots) + (b_0, b_1, \dots) \equiv (a_0 + b_0, a_1 + b_1, \dots) \quad (4.1)$$

- Multiplicación por un escalar:

$$k(a_0, a_1, \dots) \equiv (ka_0, ka_1, \dots) \quad (4.2)$$

donde k es un número real.

- **Vectores geométricos (euclidianos).** Los vectores geométricos o en breve vectores, son *segmentos orientados* en el espacio de tres dimensiones (ver [Figura 4.1](#)) que tienen las siguientes propiedades:

- Se denotarán en este libro como \vec{A} o \hat{e} (este último es un vector unitario) en lugar de usar la notación más común con letras en negrilla.
- Todo vector tiene: 1) magnitud, A , igual a la longitud (euclíadiana) del segmento correspondiente y 2) una dirección en el espacio.
- Los vectores forman con los números reales, un espacio vectorial con operaciones definidas, geométricamente, como se muestra en la [Figura 4.1](#).
- El elemento neutro de la operación suma entre vectores, es el vector nulo, que representaremos como $\vec{0}$.

- Todo vector, por definición, se puede escribir como una combinación lineal de tres vectores de una base ortonormal: $\hat{e}_0, \hat{e}_1, \hat{e}_2$. Los coeficientes de la combinación se conocen como componentes del vector:

$$\vec{A} = A_0 \hat{e}_0 + A_1 \hat{e}_1 + A_2 \hat{e}_2. \quad (4.3)$$

- El espacio de vectores es *isomórfico* a el espacio vectorial de triplets. Por la misma razón nos referiremos al vector, o bien como la entidad abstracta \vec{A} , como su representación en términos de los vectores unitarios de una base ortonormal (ver ítem anterior) o aún mejor, en términos de la tripleta:

$$\vec{A} : (A_0, A_1, A_2)$$

En este caso usaremos el símbolo “:” en lugar de “=” para dar entender que el vector *no es* una tripleta sino una entidad geométrica más abstracta.

El isomorfismo implica también, que las componentes de los vectores en las operaciones geométricas definidas en la [Figura 4.1](#), cumplen la Ecs. (4.1) y (4.2).

- Además de la suma y la multiplicación por un escalar, que caracterizan el espacio vectorial, se definen dos productos adicionales:

- **Producto escalar o producto punto**, $\vec{A} \cdot \vec{B}$. El producto escalar se define, a partir los vectores unitarios de la base, como:

$$\hat{e}_i \cdot \hat{e}_j = \delta_{ij},$$

donde δ_{ij} es el “delta de kroenecker”:

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \quad (4.4)$$

Con esta esta definición y usando la representación de los vectores dada por la Ec. (4.3) puede probarse que:

$$\vec{A} \cdot \vec{B} = A_0 B_0 + A_1 B_1 + A_2 B_2$$

- **Producto vectorial o producto cruz**, $\vec{A} \times \vec{B}$. El producto vectorial se define, a partir los vectores unitarios de la base, como:

$$\hat{e}_i \times \hat{e}_j = \epsilon_{ijk} \hat{e}_k, \quad (4.5)$$

donde ϵ_{ijk} es el “simbolo de Levi-Civita” (“levi chivita”¹⁶):

$$\epsilon_{ijk} = \begin{cases} +1 & \text{si } (i, j, k) \text{ es } (0, 1, 2), (1, 2, 0) \text{ o } (2, 0, 1) \\ -1 & \text{si } (i, j, k) \text{ es } (2, 1, 0), (0, 2, 1) \text{ o } (1, 0, 2) \\ 0 & \text{de otro modo } i = j \text{ o } j = k \text{ o } k = i \end{cases} \quad (4.6)$$

¹⁶<https://forvo.com/word/levi-civita/#it>

Al conjunto de vectores unitarios de una base que se definen cumpliendo la Ec. (4.5) se lo llama un *conjunto de vectores de mano derecha*. Con esta definición y usando la representación de los vectores dada por la Ec. (4.3) puede probarse que:

$$\vec{A} \times \vec{B} = (A_1 B_2 - A_2 B_1) \hat{e}_0 + \\ -(A_0 B_2 - A_2 B_0) \hat{e}_1 + \\ (A_0 B_1 - A_1 B_0) \hat{e}_2 \quad (4.7)$$

Está última expresión es tan elaborada que con frecuencia se usa la regla mnemotécnica:

$$\vec{A} \times \vec{B} = \begin{vmatrix} \hat{e}_0 & \hat{e}_1 & \hat{e}_2 \\ A_0 & A_1 & A_2 \\ B_0 & B_1 & B_2 \end{vmatrix} \quad (4.8)$$

Donde $|M|$ es el determinante de la matriz M .

- Otras identidades útiles:
 - Propiedad cíclica del **triple producto escalar**:

$$\vec{A} \cdot (\vec{B} \times \vec{C}) = \vec{C} \cdot (\vec{A} \times \vec{B}) = \vec{B} \cdot (\vec{C} \times \vec{A}) \quad (4.9)$$

- **Triple producto vectorial**:

$$\vec{A} \times (\vec{B} \times \vec{C}) = (\vec{A} \cdot \vec{C})\vec{B} - (\vec{A} \cdot \vec{B})\vec{C} \quad (4.10)$$

Algoritmos para conjuntos y tuplas

Todos los lenguajes modernos de programación, definen tipos especiales para representar conjuntos y tuplas. En Python existen tres tipos de objetos básicos para este propósito: *listas*, *tuplas* y *diccionarios*. Existen sútiles diferencias entre las listas y las tuplas en Python y en general usaremos con más frecuencia las primeras. Para los algoritmos de este libro, es importante entender las *operaciones* entre listas, que son diferentes a las operaciones en el espacio vectorial de las tuplas matemáticas que definimos antes.

Así, por ejemplo, en el siguiente algoritmo se construye una lista con las componentes del vector de estado de una partícula, “sumando” las listas de las componentes de su vector posición y velocidad:

```
#Lista de componentes del vector posición
r=[1,0,3]
#Lista de componentes del vector velocidad
v=[0,-1,0]
#Lista de componentes del vector de estado
X=r+v
```

```
X = [1, 0, 3, 0, -1, 0]
```

El operador `+`, entre listas y tuplas de Python produce la unión de los elementos de las listas.

Usando este operador se pueden hacer algoritmos prácticos como el que se muestra a continuación:

```
def f(x):
    from math import sin
    y=sin(x)/x
    return y

valores_de_x=[1.0,2.0,3.0]
valores_de_f=[]
for x in valores_de_x:
    valores_de_f+=[f(x)]
```

valores de f = [0.8414709848078965, 0.45464871341284085, 0.0470400026866224]

aquí, comenzamos con un conjunto vacío, `valores_de_f=[]` y después, dentro de un ciclo, usamos el operador de acumulación `+=` para agregar elementos al conjunto. Este es un método muy común usado en el lenguaje para construir “tablas de valores”, que pueden, por ejemplo usarse para hacer gráficos de funciones.

Algoritmos para vectores

Los vectores forman un “capítulo” en la computación separado de las listas y las tuplas. La razón básica son sus propiedades matemáticas y las operaciones definidas entre ellos. Bibliotecas de rutinas muy completas existen en todos los lenguajes de programación para representar este tipo de entidades matemáticas.

En Python y a lo largo de este libro usaremos los objetos y rutinas de los paquetes NumPy y SPICE para definir y manipular vectores.

El ejemplo abajo muestra como se calcula el ángulo entre dos vectores θ_{AB} , a partir de la interpretación geométrica del producto punto $\vec{A} \cdot \vec{B} = AB \cos \theta_{AB}$ (ver problemas al final del capítulo):

```
#Definimos los vectores
from numpy import array
A=array([1.0,0.0,2.0])
B=array([0.0,1.0,3.0])

#Calculamos el producto escalar y vectorial
from numpy import dot
ApuntoB=dot(A,B)

#El ángulo entre los vectores
from numpy import arccos
from numpy.linalg import norm
anguloAB=arccos(ApuntoB/(norm(A)*norm(B)))
```

AnguloAB = 31.948059431330062 grados



Nota

Radianes y grados en los algoritmos. Es importante entender que las funciones trigonométricas inversas como `arccos`, devuelven, en todos los lenguajes de programación, valores de los ángulos en radianes. En el caso anterior, por ejemplo, el valor de la variable `anguloAB` al final del algoritmo en realidad era 0.5575988266995369. Sin embargo, decidimos mostrar su valor en grados después de multiplicar `anguloAB` por el factor de conversión $\pi/180$ (esta operación no se muestra en el código.) Así lo seguiremos haciendo en el resto del libro. El lector que use los algoritmos no debe olvidar multiplicar por el factor de conversión para reconstruir los resultados mostrados aquí.

Un procedimiento similar, esta vez usando vectores y rutinas de SPICE (internamente SPICE usa vectores o arreglos de NumPy), puede usarse para calcular el triple producto vectorial:

```
from numpy import array
from spiceypy import vdot,vcrss

A=array([2.0,2.0,1.0])
B=array([0.0,-1.0,0.0])
C=array([0.0,0.0,2.0])

AxBxC=vdot(A,B)*C-vdot(A,C)*B
```

$$A \times (B \times C) = [-0. \quad 2. \quad -4.]$$

También podemos usarlas para verificar la propiedad cíclica del triple producto escalar:

```
from numpy import array
from spiceypy import vnrm,vdot,vcrss
A=array([2.0,2.0,1.0])
B=array([0.0,-1.0,0.0])
C=array([0.0,0.0,2.0])

ABC=vdot(A,vcrss(B,C))
CAB=vdot(C,vcrss(A,B))
BCA=vdot(B,vcrss(C,A))
```

$$\begin{aligned} A \cdot (B \times C) &= -4.0 \\ C \cdot (A \times B) &= -4.0 \\ B \cdot (C \times A) &= -4.0 \end{aligned}$$

Con lo que se verifica la identidad (al menos para los vectores escogidos.)

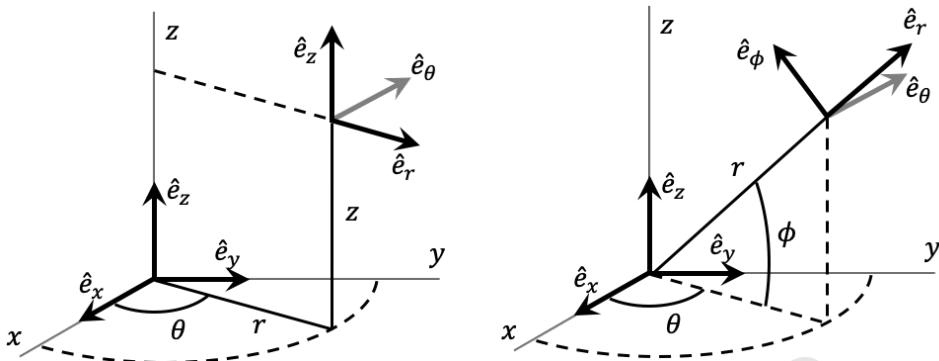


Figura 4.2: Definición de los sistemas de coordenadas usados en este texto

4.1.2. Sistemas de coordenadas

A lo largo de este libro, usaremos los tres sistemas de coordenadas ortogonales clásicos (cartesianas, cilíndricas y esféricas, ver [Figura 4.2](#)) con algunas convenciones más propias de la astronomía y la mecánica celeste que del cálculo.

A continuación, y en especial para clarificar nuestra notación, enumeramos detalladamente las propiedades de cada sistema.

- **Sistema de coordenadas cartesiano.**

- Coordenadas: $x \in (-\infty, +\infty)$, $y \in (-\infty, +\infty)$, $z \in (-\infty, +\infty)$.
- Vectores unitarios: $\hat{e}_x, \hat{e}_y, \hat{e}_z$.
- Comentarios:
 - En todos los casos la orientación de los ejes obedecerá la *regla de la mano derecha*, es decir, los sistemas cartesianos usados en el texto y cuyos ejes están definidos por el conjunto de vectores unitarios ($\hat{e}_x, \hat{e}_y, \hat{e}_z$) forman un *conjunto de mano derecha* (ver Ec. 4.5), a saber, en forma explícita:

$$\begin{aligned}\hat{e}_x \times \hat{e}_y &= \hat{e}_z \\ \hat{e}_y \times \hat{e}_z &= \hat{e}_x \\ \hat{e}_z \times \hat{e}_x &= \hat{e}_y\end{aligned}\tag{4.11}$$

- **Sistema de coordenadas cilíndrico** (ver [Figura 4.2](#)).

- Coordenadas: $r \in [0, +\infty)$, $\theta \in [0, 2\pi)$, $z \in (-\infty, +\infty)$.
- Conversión al sistema de coordenadas cartesianas:

$$\begin{aligned}x &= r \cos \theta \\ y &= r \sin \theta\end{aligned}\tag{4.12}$$

- Vectores unitarios expresados en el sistema de coordenadas cartesianas:

$$\begin{aligned}\hat{e}_r &= \cos \theta \hat{e}_x + \sin \theta \hat{e}_y \\ \hat{e}_\theta &= -\sin \theta \hat{e}_x + \cos \theta \hat{e}_y \\ \hat{e}_z &= \hat{e}_z\end{aligned}\quad (4.13)$$

- Comentarios:

- El conjunto de vectores unitarios ($\hat{e}_r, \hat{e}_\theta, \hat{e}_z$) forman un conjunto de mano derecha tal y como se definió en las Ecs. (4.11).
- Nótese que, a diferencia de la notación usada generalmente en los textos de cálculo, la coordenada cilíndrica r usa la misma letra que la coordenada esférica r y la magnitud del vector posición (ver siguiente sección). La distinción entre las tres, dependerá del contexto.
- Usaremos la letra griega θ para denotar el ángulo *acimutal*, a diferencia de la notación convencional que usa esta letra para la coordenada esférica polar (ángulo del vector posición respecto al eje z.)

- Sistema de coordenadas esférico (ver Figura 4.2).

- Coordenadas: $r \in [0, +\infty), \theta \in [0, 2\pi), \phi \in [-\pi/2, +\pi/2]$.
- Conversión al sistema de coordenadas cartesianas:

$$\begin{aligned}x &= r \cos \phi \cos \theta \\ y &= r \cos \phi \sin \theta \\ z &= r \sin \phi\end{aligned}\quad (4.14)$$

- Vectores unitarios expresados en el sistema de coordenadas cartesianas:

$$\begin{aligned}\hat{e}_r &= \cos \phi \cos \theta \hat{e}_x + \cos \phi \sin \theta \hat{e}_y + \sin \phi \hat{e}_z \\ \hat{e}_\theta &= -\sin \theta \hat{e}_x + \cos \theta \hat{e}_y \\ \hat{e}_\phi &= \cos \phi \cos \theta \hat{e}_x + \sin \phi \sin \theta \hat{e}_y - \sin \phi \hat{e}_z\end{aligned}\quad (4.15)$$

- Comentarios:

- El conjunto de vectores unitarios ($\hat{e}_r, \hat{e}_\theta, \hat{e}_\phi$) forman un conjunto de mano derecha tal y como se definió en las Ecs. (4.11).
- Nótese que, a diferencia de la notación usada generalmente en los textos de cálculo, la coordenada esférica ϕ se medirá respecto al plano $x - y$ (como una *latitud*) en lugar de hacerlo respecto al eje z (como una *colatitud*).

Una interesante página interactiva que permite visualizar mejor la definición de los sistemas de coordenadas y la orientación de los vectores coordenadas puede encontrarse en los siguientes enlaces, tanto para el [sistema de coordenadas cilíndrica](#)¹⁷ como para el [sistema de coordenadas esféricas](#)¹⁸.

¹⁷<http://dynref.engr.illinois.edu/rvy.html>

¹⁸<http://dynref.engr.illinois.edu/rvy.html>

4.1.3. Funciones

Una función es, en términos informales, una regla de correspondencia que asocia los elementos de un conjunto de partida o *dominio* (p.e. el conjunto de los números reales \mathbb{R} , el conjunto de puntos en un plano \mathbb{R}^2 o de eventos en el espacio tiempo \mathbb{R}^4) con los de otro conjunto, llamado rango, de modo a que cada elemento del dominio le corresponde **uno y solo un elemento del rango**.

Entre los distintos tipos de funciones que reconoce el análisis matemático, en este libro nos concentraremos en:

- **Funciones de variable real:** Dominio y rango \mathbb{R} . Ejemplo: $f(t) = t^2$.
- **Funciones de muchas variables o Campos escalares:** Dominio \mathbb{R}^n , rango \mathbb{R} . Ejemplos: $f(x, y) = x^2 + y^2$, $H(\{q_i\}_N) = \sum_i q_i^2$ (para la notación del conjunto $\{q_i\}$ ver la Sección 4.1.1).
- **Funciones vectoriales:** Dominio \mathbb{R} , rango \mathbb{R}^n . Ejemplo: $\vec{u}(t) = kr^{-n}\hat{\ell}_r$.
- **Funciones vectoriales de muchas variables o Campos vectoriales:** Dominio \mathbb{R}^n , rango \mathbb{R}^3 . Ejemplo: $\vec{F}(r, \theta, z) : -k(r \cos \theta, r \sin \theta, z)$.

Nota

t como variable genérica de las funciones. En todos los textos de matemáticas (incluso en los de física) se acostumbra usar x como el nombre preferido para representar, de forma genérica, la variable independiente de las funciones. En lo sucesivo cambiaremos esta convención al llamar t a la variable independiente genérica. La razón no puede ser más sencilla: en la mecánica t es el nombre que damos a la variable independiente por excelencia, el tiempo, de modo que muchas de las fórmulas que desarrollaremos en este capítulo, se trasladaran simbólicamente casi sin modificación a la mecánica.

Es obvio que la elección de la letra con la que representamos la variable independiente, no modifica en nada las definiciones y teoremas que veremos en esta sección, de modo que esperamos esta elección no moleste a los más conservadores ni confunda a quienes han estudiado ampliamente estos temas en otros textos.

Algoritmos para funciones

Hay dos maneras de definir una función en Python: 1) como una rutina o 2) como una función `lambda`.

Como una rutina, una función en Python puede recibir como “argumentos” de entrada no solo las variables de la función sino también argumentosopcionales.

La siguiente función, por ejemplo, permite calcular el valor de la energía potencial de un sistema físico usando la función de varias variables $U(\vec{r}) = kr^n$ (siendo $\vec{r} : x, y, z$ el vector posición y r su magnitud.)

(Algoritmo 4.1)

```
def U(x,y,z,k=1,n=-1):
    r=(x**2+y**2+z**2)**0.5
    return k*r**n
```

$U(1.0, 2.0, 0.0)$ con $k = 1$ y $n = -1$ (valores por defecto) = 0.4472135954999579
 $U(1.0, 2.0, 0.0)$ con $k = 6.67e-11$ y $n = -2$ = 1.334e-11

Nota

Argumentos obligatorios y argumentos opcionales. Toda rutina en Python puede tener unos argumentos obligatorios (que llamaremos variables) o unosopcionales.

Las variables son en estricto sentido una tupla de valores, por ejemplo x, y, z en la función U en el Alg. (4.1).

Los argumentosopcionales son, por otro lado, un diccionario de valores, que no es otra cosa que una lista de valores identificados con un nombre (también llamado clave o key). En la función U en el Alg. (??) los argumentosopcionales son $k=1, n=-1$.

En Python las variables y las opciones de una rutina pueden representarse usando los objetos especiales `*variables` y `**opciones`. El uso de estos objetos especiales no es muy común, pero en ciertas situaciones puede ser bastante útil.

Una forma alternativa de la rutina para U en el Alg. (??) es:

```
def U(*variables,**opciones):
    x,y,z=variables
    r=(x**2+y**2+z**2)**0.5
    return opciones["k"]*r**opciones["n"]
```

que se puede invocar usando:

```
var=1.0,2.0
opc=dict(k=1,n=-2)
U(*var,**opc)
```

No parece muy práctico, pero como veremos puede ser muy útil en ciertas situaciones especiales.

Las funciones `lambda` se usan para representar funciones muy abreviadas y no tienen argumentos distintos de las variables de las que dependen.

Así, por ejemplo, el siguiente algoritmo define una función `lambda`, U_x , basada en la función U del Alg. (??), que depende solo de la variable x cuando y asumes constante los valores de y y z (U_x será util para calcular más abajo la derivadas parcial de U respecto a x):

(Algoritmo 4.2)

```

y=1.0
z=1.0
k=1
n=-2
U_x=lambda x:U(x,y,z,k,n)

```

$U_x(0.0) = 0.4999999999999999$

4.1.4. Derivadas

La derivada de una función de variable real es en sí misma una función definida por el límite:

$$\frac{df}{dt} \equiv \lim_{\Delta t \rightarrow 0} \frac{f(t) - f(t + \Delta t)}{\Delta t} \quad (4.16)$$

Si el límite no existe decimos que la función no es derivable en t .

Nota

Notación de la derivada. A lo largo de la historia la manera como se ha representado la función derivada ha cambiado. Existen al menos tres notaciones comunes:

- La **notación de Leibniz**, df/dt , d^2f/dt^2 . En esta notación la derivada se representa como si fuera la razón entre dos cantidades, pero no es así ¡mucho cuidado! Usaremos la notación de Leibniz especialmente para representar la derivada de funciones que se escriben de forma explícita, así por ejemplo:

$$\frac{d}{dt} \left(\frac{1}{2} t^2 \right)$$

- La **notación de Newton**, \dot{f} , \ddot{f} . Esta será la forma que usaremos para denotar a lo largo del libro las derivadas respecto del tiempo (o el tiempo propio en relatividad).
- La **notación de Lagrange**, f' , f'' , $f^{(n)}$.
- La **notación de Euler**, Df , D^2f , $D^n f$, que no usaremos aquí pero es la notación menos común, pero puede aparecer en el contexto de la mecánica de fluidos.

La definición de derivada de las funciones de variable real como un límite, se extiende por analogía a campos escalares, funciones vectoriales o campos vectoriales. Para las funciones que dependen de varias variables, sin embargo, se usa

una notación y un nombre diferente: **derivada parcial**. La derivada parcial de un campo escalar se define como:

$$\frac{\partial f}{\partial q_k} = \lim_{\Delta q_k \rightarrow 0} \frac{f(q_1, q_2, \dots, q_k + \Delta q_k, \dots, q_N) - f(q_1, q_2, \dots, q_k, \dots, q_N)}{\Delta q_k}$$

La derivada parcial se calcula de la misma manera que la derivada de una variable, con la salvedad de que al hacerlo se asume que todas las demás variables de la función son constantes.

En muchas partes en este libro, y por economía usaremos la notación de Euler para las derivadas parciales, a saber:

$$\partial_x f \equiv \frac{\partial f}{\partial x}$$

En esta notación una derivada parcial múltiple se escribirá como:

$$\partial_{xyz} f(x, y, z) \equiv \frac{\partial^3 f}{\partial x \partial y \partial z}$$

A pesar de que la derivada parcial tiene una definición *numérica* análoga a la de la derivada total, existe una sutil diferencia entre ambas.

Imagine que tenemos una **variable independiente** t y definimos, a partir de ella, una nueva variable u que es función de t (variable dependiente).

¿Cómo podemos calcular la derivada de una función de la nueva variable $f(u)$ respecto de la variable independiente t ?

Proposición 4.1

Regla de la Cadena. Dada una función compuesta $f(u(t))$, la derivada de f respecto a t es:

$$\frac{df}{dt} = \frac{df}{du} \frac{du}{dt}$$

Decimos que la función f depende *implícitamente* de la variable independiente t . En este sentido la regla de la cadena es una regla de *derivación implícita*.

Usando la notación de Newton la expresión anterior se escribirá de forma abreviada:

$$\dot{f}(t) = \dot{u} \frac{df}{du}$$

¿Qué pasa en el caso en el que f depende de varias variables dependientes, por ejemplo $f(q_1(t), q_2(t), \dots, q_N(t)) \equiv f(\{q_i(t)\}_N)$?

En este caso la regla de la cadena se puede generalizar como:

$$\frac{df}{dt} = \frac{\partial f}{\partial q_1} \frac{dq_1}{dt} + \frac{\partial f}{\partial q_2} \frac{dq_2}{dt} + \dots + \frac{\partial f}{\partial q_N} \frac{dq_N}{dt} = \sum_i \frac{\partial f}{\partial q_i} \frac{dq_i}{dt} = \sum_i \dot{q}_i \partial_{q_i} f$$

Ahora bien: ¿existirá, en este caso, la derivada parcial de f respecto de t ?

La respuesta a esta pregunta, ilustra, justamente, la diferencia sutil entre la derivada ordinaria o *derivada total* d/dt y la derivada parcial $\partial/\partial t$.

Hay dos situaciones posibles:

- Si la función f no depende explícitamente de t , es decir si la variable t no aparece en la fórmula de f , entonces $\partial f / \partial t = 0$. Este resultado es *independiente* de que f dependa implícitamente de t a través de otras variables dependientes.
- Ejemplo:** si $f(q, t) = q^2$, entonces: $\partial f / \partial t = 0$ aunque, por regla de la cadena, $df/dt = 2q\dot{q}$.
- Si la fórmula de la función f contiene la variable t , entonces su derivada parcial puede ser distinta de cero (dependiendo de la forma funcional de f).
- Ejemplo:** Si $f(q, t) = q^2 + \sin t$, entonces: $\partial f / \partial t = \cos t$ y $df/dt = 2q\dot{q} + \cos t$.

En este sentido la derivada parcial es como un “operador semántico”, es decir un operador sobre las “letras” que aparecen en la fórmula de la función.

Teniendo en cuenta esta propiedad, la forma más general de la regla de la cadena, para una función de varias variables (campo escalar o vectorial) será:

$$\frac{d}{dt}f(\{q_i\}, t) = \sum_i \dot{q}_i \partial_{q_i} f + \frac{\partial f}{\partial t} \quad (4.17)$$

4.1.5. Funciones homogéneas

Existe un interesante conjunto de funciones para las cuales hay una relación no trivial entre su derivada y el valor de la función misma. Se conocen como **funciones homogéneas**:

Definición 4.1

Funciones homogéneas. Una función general $f(\{q_i\})$ se llama homogénea si frente a una operación de escalado de sus variables (multiplicación por un escalar), la función *escala* también. En términos matemáticos:

$$f(\{\lambda q_i\}) = \lambda^k f(\{q_i\})$$

donde λ es un número real y k se conoce como el **orden** de la función.

Las funciones homogéneas son, generalmente polinomios y funciones racionales. Así por ejemplo $f(x, y) = x^2/a^2 + y^2/b^2$, con a y b constantes, y que representa la ecuación algebraica de una elipse, es una función homogénea de grado $k = 2$. De otro lado $f(x) = x^3y^2 + y^5$ es homogénea de grado $k = 5$.

Las funciones homogéneas más interesantes para nosotros en este libro son del tipo $f(\vec{r}) = kr^n$ que son homogéneas de grado $k = n$ (ver problemas al final del capítulo.)

Como mencionamos desde el principio, las derivadas de las funciones homogéneas tienen una propiedad muy importante:

Proposición 4.2

Teorema de funciones homogéneas de Euler. Si una función $f(\{q_i\}_N)$ es homogénea de grado k , entonces:

$$\sum_i q_i \frac{\partial f}{\partial q_i} = kf$$

Para funciones homogéneas definidas en el espacio de tres dimensiones, el teorema de Euler se puede escribir como:

$$\vec{r} \cdot \vec{\nabla} f = kf$$

4.1.6. Derivada vectorial

Para funciones de varias variables (especialmente aquellas con dominio en el espacio coordenado \mathbb{R}^3) se definen generalizaciones vectoriales de la derivada que tienen motivaciones e interpretaciones geométricas específicas.

El *operador diferencial vectorial* básico se conoce como el **gradiente**. Denotado comúnmente como $\vec{\nabla}$, en coordenadas cartesianas se define explícitamente como:

$$\vec{\nabla} f(x, y, z) = \frac{\partial f}{\partial x} \hat{e}_x + \frac{\partial f}{\partial y} \hat{e}_y + \frac{\partial f}{\partial z} \hat{e}_z \quad (4.18)$$

El operador gradiente en el sistema de coordenadas cilíndrico (con la notación definida anteriormente) está dado por:

$$\vec{\nabla} f(r, \theta, z) = \frac{\partial f}{\partial r} \hat{e}_r + \frac{1}{r} \frac{\partial f}{\partial \theta} \hat{e}_\theta + \frac{\partial f}{\partial z} \hat{e}_z \quad (4.19)$$

Donde el factor $1/h_\theta \equiv 1/r$ se conoce como *factor de escala*.

Por su parte en coordenadas esféricas (con la notación definida anteriormente):

$$\vec{\nabla} f(r, \theta, \phi) = \frac{\partial f}{\partial r} \hat{e}_r + \frac{1}{r} \frac{\partial f}{\partial \theta} \hat{e}_\theta + \frac{1}{r \cos \phi} \frac{\partial f}{\partial \phi} \hat{e}_\phi \quad (4.20)$$

En este caso se ha introducido un nuevo factor de escala: $h_\phi \equiv r \cos \phi$.

Nota

Una notación para el gradiente. Como lo hicimos con la derivada parcial, a lo largo de este libro, abreviaremos el gradiente usando la notación especial:

$$\partial_{\vec{r}} f \equiv \frac{\partial f}{\partial \vec{r}} \equiv \vec{\nabla} f$$

Aunque no es una notación muy rigurosa, permite abreviar expresiones que de otra manera serían muy elaboradas. Así por ejemplo, la re-

Nota (Cont.)

gla de la cadena (Ec. 4.17) para funciones definidas en el espacio coordenado, se puede escribir de forma compacta como:

$$\dot{f}(x, y, z, t) = \partial_{\vec{r}} f \cdot \dot{\vec{r}} + \partial_t f \quad (4.21)$$

Existen otros operadores vectoriales (laplaciano, divergencia, rotacional) sobre los que no profundizaremos aquí por no ser de mucha utilidad práctica en la mecánica celeste (al menos no al nivel de este libro.)

Algoritmos para la derivada

Existen diversos algoritmos para calcular la derivada de una función en una o varias variables. En este libro, en donde sea necesario, nos apoyaremos de la biblioteca científica `scipy` y su rutina `derivative` que permite calcular, numéricamente, derivadas de cualquier orden.

El siguiente algoritmo ilustra el uso de `derivative` y sus opciones:

```
def f(t):
    from math import sin
    return sin(t)/t

#Valor de la variable independiente donde queremos la derivada
t=2.0

from scipy.misc import derivative

#Primera derivada usando un dx=0.01 y 3 puntos
dfdt=derivative(f,t,dx=1e-2,n=1,order=3)

#Segunda derivada en t
d2fdt2=derivative(f,t,dx=1e-2,n=2,order=5)

dfdt : Numérica = -0.4353938258295498, Exacta = -0.43539777497999166
d2fdt2 : Numérica = -0.019250938436687903, Exacta = -0.01925093843284925
```

Usando `derivative` es posible diseñar funciones para calcular derivadas parciales e incluso gradientes (para los cuales no existen funciones en la biblioteca `scipy`). Así por ejemplo:

```
def f(x,y,z):
    from math import sin
    return sin(x*y*z)/(x*y*z)

def partial_derivative_x(f,x,y,z,**opciones):
    f_solo_x=lambda x:f(x,y,z)
    dfdx=derivative(f_solo_x,x,**opciones)
    return dfdx
```

```
x=1.0
y=2.0
z=3.0
dfdx=partial_derivative_x(f,x,y,z,dx=0.01)
```

`dfdx: Numérica = 1.0061803563982654, Exacta = 1.006739536350187`

Nótese como usamos aquí la función `lambda f_solo_x`, de la manera que lo hicimos en el Algoritmo (4.2) para conseguir el resultado deseado.

4.1.7. Integrales

Se llama **antiderivada** de una función de variable real $f(t)$, a la función $F(t)$ cuya derivada es igual a la función original:

$$\dot{F}(t) = f(t)$$

O en notación *integral*:

$$F(t) \equiv \int f(t) dt$$

A $F(t)$ o equivalentemente $\int f(t) dt$ se la llama también la **integral indefinida** de $f(t)$.

La antiderivada permite calcular la **cuadratura de una función**, que no es otra cosa que el área encerrada por la curva en el plano cartesiano definido por la variable independiente y los valores de la función:

Proposición 4.3

Fórmula de Newton-Leibniz.^a Dada una función $f(t)$ que tiene antiderivada $F(t)$ definida en el intervalo $[a, b]$, el área o cuadratura de la función en el mismo intervalo está dado por:

$$\int_a^b f(t) dt = F(b) - F(a)$$

A la cantidad $\int_a^b f(t) dt$ se la llama **integral definida** de $f(t)$.

^aA esta fórmula se la llama a menudo *segundo teorema fundamental del cálculo*

En términos de la integral definida podemos definir una nueva función:

$$I(t) = \int_a^t f(\tau) d\tau$$

Nótese que para ser rigurosos hemos cambiado el nombre de la “variable de integración” τ para no confundirla con el límite superior de la integral t .

Esta nueva función tiene una importante propiedad:

Proposición 4.4

Teorema fundamental del cálculo. Dada una función $f(t)$ integrable en el intervalo $[a, b]$, si definimos la función $I(t) = \int_a^t f(\tau) d\tau$, entonces:

$$\frac{dI}{dt} = f(t)$$

o bien,

$$\frac{d}{dt} \int_a^t f(\tau) d\tau = f(t) \quad (4.22)$$

Es interesante anotar que aunque la antiderivada $F(t)$ y la función $I(t)$ tienen la misma derivada en t , es decir $dF/dt = dI/dt = f(t)$, no son necesariamente la misma función. Considere, por ejemplo, el hecho elemental de que $I(a) = 0$ (por definición) mientras que $F(a)$ podría ser cualquier número (incluyendo cero por supuesto.)

4.1.8. Integrales vectoriales

Una extrapolación del concepto de integral a funciones de varias variables (campos escalares y campos vectoriales) conduce a algunas operaciones integrales de gran importancia en la física. Para los propósitos de lo que veremos en este libro, son de particular interés las integrales del tipo:

$$\int \vec{F} \cdot d\vec{r},$$

que se define sobre todos los valores de \vec{r} de una curva en el espacio coordenado. A esta integral se la conoce como **integral de línea**. Si la trayectoria es cerrada, escribiremos:

$$\oint \vec{F} \cdot d\vec{r},$$

que no se diferencia (matemáticamente) en nada de una integral de línea. A esta integral la llamaremos **circulación** del campo vectorial \vec{F} .

Otro tipo de integral vectorial de interés es:

$$\int_{\Sigma} \vec{F} \cdot d\vec{S}$$

Donde $d\vec{S}$ tiene dirección normal a la superficie Σ (formada por el lugar geométrico de todos los puntos que la definen) y magnitud igual al área de una fracción infinitesimal de la superficie.

Proposición 4.5

teorema de Stokes. Si $\vec{F}(\vec{r})$ es un campo vectorial diferenciable en todos los puntos del espacio, entonces:

Proposición 4.5 (Cont.)

$$\oint \vec{F} \cdot d\vec{r} = \int_{\Sigma} (\vec{\nabla} \times \vec{F}) \cdot d\vec{S}$$

Donde Σ es cualquier superficie que tenga como frontera la trayectoria sobre la que se define la circulación.

Un importante corolario del teorema de Stokes es el siguiente:

Proposición 4.6

Corolario de Stokes. Si el campo vectorial $\vec{F}(\vec{r})$ tiene circulación nula:

$$\oint \vec{F} \cdot d\vec{r} = 0$$

Entonces existe un campo escalar $U(\vec{r})$ tal que:

$$\vec{F} = \vec{\nabla} U$$

Llamamos a U la función *potencial* de \vec{F} .

Algoritmos para la integral

El cálculo numérico de integrales es una basta área del análisis numérico. En cada lenguaje de programación es posible encontrar bibliotecas completas con rutinas para el cálculo de aproximaciones numéricas de integrales definidas e integrales vectoriales.

Para los propósitos de este libro, usaremos la rutina `quad` de la biblioteca SciPy para calcular numéricamente integrales definidas de funciones de variable real.

En el algoritmo provisto a continuación, calculamos, por ejemplo, el trabajo $W \equiv \int F(x) dx$ sobre una partícula que se mueve en una dimensión sometida a una fuerza del tipo $F(x) = -kx$, asumiendo que $k = 0,1$ y que la partícula se desplaza entre $x = 1,0$ y $x = 5,0$:

```
#El integrando debe definirse como una rutina
def F(x,k=1):
    return -k*x

from scipy.integrate import quad
k=0.1
x0=1.0
x1=5.0
integral=quad(F,x0,x1,args=(k,))
```

Integral: Numérica = (-1.2, 1.3322676295501878e-14), Exacta = -1.2

Nótese que los argumentos opcionales del integrando se pasan como la tupla `args` que en este caso, dado que la función solo depende de un parámetro opcio-

nal, se escribe de forma poco intuitiva como `args=(k,)` donde la coma final es obligatoria.

El resultado de la función quad es una tupla con dos números: el valor de la integral y el error estimado de la misma. Como vemos, en el ejemplo arriba, la integral es prácticamente exacta.

Nota

Cuadraturas Gaussianas. El método usado por quad para calcular la integral se conoce como *cuadraturas gaussianas* y approxima la integral como una serie de pocos términos del valor de la función definido en algunos puntos específicos [7]. Las cuadraturas gaussianas permiten calcular la integral de funciones polinómicas de forma *exacta*. Esta es la razón por la cual la integral en el ejemplo dado aquí, es idéntica al valor esperado.

4.1.9. Ecuaciones diferenciales

Encontrar la antiderivada de una función (ver [Sección 4.1.7](#)), se puede formular, de forma general, como el problema de encontrar una función $F(t)$ tal que:

$$\frac{dF(t)}{dt} = f(t) \quad (4.23)$$

La solución a este problema es, por definición:

$$F(t) = \int f(t) dt$$

La integral indefinida en el lado derecho de la anterior ecuación y los métodos numéricos o exactos (analíticos) para obtenerla (no cubiertos en este corto resumen) representan unas de las herramientas matemáticas más útiles de la física.

Pero, existen situaciones en las que el cálculo de una antiderivada no se reduce simplemente a una integral indefinida. Considere por ejemplo el siguiente problema:

$$\frac{d^2F(t)}{dt^2} = -kF(t) \quad (4.24)$$

que, en palabras, se formularía como: encontrar la función cuya segunda derivada es proporcional (k se supone constante) al negativo de ella misma.

Ambas, las Ecs. (4.23) y (4.24) se conocen como **ecuaciones diferenciales**.

Las ecuaciones diferenciales se clasifican según:

- **Su orden.** El orden de una ecuación diferencial es igual al máximo orden de la derivada de la función objetivo (antiderivada) que aparece en la ecuación. La ecuación diferencial básica (4.23) es, por ejemplo, de *primer orden* porque solo involucra la *primera* derivada de la función $F(t)$. Por su parte, la ecuación diferencial (4.24) es una ecuación diferencial de *segundo orden*.

- **Su linealidad.** Una ecuación diferencial que solo depende de primeras potencias de la función y sus derivadas se dice que es lineal. En caso contrario tenemos una *ecuación diferencial no lineal*. Las ecuaciones (4.23) y (4.24) son lineales, pero la siguiente ecuación diferencial de primer orden, no lo es:

$$\frac{dF(t)}{dt} = \frac{h}{F(t)},$$

donde h es una constante.

- **El número de variables independientes.** Una ecuación diferencial en la que la función depende de una sola variable real se conoce como una **ecuación diferencial ordinaria** (*ODE* por la sigla en inglés de *ordinary differential equation*). Si, por otro lado, la función es un campo escalar o vectorial y la ecuación diferencial se expresa en términos de derivadas parciales (y totales) hablamos de una **ecuación diferencial parcial** (*PDE* por sus siglas en inglés).
- **El número de funciones o variables dependientes.** Es posible que un problema implique encontrar más de una antiderivada. En ese caso hablamos de un **sistema de ecuaciones diferenciales**. Un caso común de sistemas de ecuaciones diferenciales se produce cuando queremos encontrar la antiderivada de una función vectorial (cada componente de una función vectorial es una función en sí misma). El caso más importante en la física de un sistema de ecuaciones diferenciales es la *ecuación de movimiento de una partícula* (que exploraremos a fondo en la ??):

$$\frac{d^2\vec{r}(t)}{dt^2} = \vec{a}$$

Esta ecuación es una forma abreviada de escribir el sistema de ecuaciones diferenciales:

$$\begin{aligned}\frac{d^2x(t)}{dt^2} &= a_x \\ \frac{d^2y(t)}{dt^2} &= a_y \\ \frac{d^2z(t)}{dt^2} &= a_z\end{aligned}\tag{4.25}$$

Como las cantidades a_x , a_y y a_z pueden ser a su vez funciones del tiempo, de las funciones x , y , z y de sus derivadas, se habla, además, de un **sistema de ecuaciones diferenciales acopladas**.

- **Las condiciones que deben proveerse para resolverla.** La solución abstracta de una ecuación diferencial, es decir, el problema de encontrar la antiderivada general, es el equivalente a la integral indefinida. Las integrales definidas, por su lado, equivalen en la teoría de ecuaciones diferenciales a los que se

conocen como **problemas de valor inicial** (*IVP* por el acrónomimo en inglés de *initial value problem*.) En este tipo de problemas la solución a la ecuación diferencial consiste en encontrar el valor de la función para cualquier valor de la variable independiente una vez se ha provisto el valor de la función (o funciones) y de sus derivadas, en un valor específico o inicial de la variable independiente. Así por ejemplo:

$$\frac{d^2\vec{r}(t)}{dt^2} = \vec{a}, \text{ con } \vec{r}(0) : (0, 0, 0) \text{ y } \dot{\vec{r}}(0) : (1, 0, 0)$$

es un IVP.

Por otro lado un **problema de condiciones de frontera** (*BVP* por la sigla en inglés de *boundary value problem*) es aquel en el que el valor de la función dependiente (no de sus derivadas necesariamente) se provee para varios valores de la variable independiente. Así por ejemplo:

$$\frac{d^2F(t)}{dt^2} = -F(t), \text{ con } F(0) = 0 \text{ y } F(\pi/2) = 1,0,$$

es un BVP.

Como se intuye fácilmente, la dificultad en la solución a una ecuación diferencial, como sucede también con las ecuaciones algebraicas, puede aumentar con su orden. Sin embargo, usando variables auxiliares, siempre es posible escribir una ecuación diferencial de orden M como un sistema de M ecuaciones diferenciales de primer orden.

Por ejemplo, si en la Ec. (4.24) llamamos $G(t) \equiv dF(t)/dt$, esa ecuación diferencial de segundo orden se puede escribir como el sistema de ecuaciones diferenciales:

$$\frac{dF}{dt} = G \tag{4.26}$$

$$\frac{dG}{dt} = -kF \tag{4.27}$$

A esta sistema de ecuaciones, lo llamamos el *sistema de ecuaciones diferenciales reducido*.

La reducción del orden será un método muy utilizado en este libro para abordar la solución a las ecuaciones diferenciales de la mecánica celeste y analítica.

Algoritmos para la solución de ODE

La solución aproximada de ecuaciones diferenciales es una de las áreas de mayor interés en el análisis numérico. Sus beneficios prácticos se extienden desde la física teórica y la economía hasta la climatología y la simulación del vuelo de aviones y vehículos espaciales. A lo largo de los últimos 350 años (y en paralelo con la evolución de la mecánica), se han desarrollado métodos numéricos para aproximar

la solución de todos los tipos de ecuaciones diferenciales que hemos mencionado hasta aquí.

En este libro, sin embargo, nos concentraremos en la solución de sistemas ecuaciones diferenciales ordinarias con valores iniciales o *IVP*.

Los métodos numéricos generales, desarrollados para resolver este tipo de problemas (ver [7] para detalles sobre los métodos y algoritmos explícitos), suponen que la ecuación o sistema de ecuaciones diferenciales que queremos resolver puede escribirse como un sistema reducido de ecuaciones diferenciales de primer orden de la forma:

$$\{\dot{Y}_i = f_i(\{Y_k\}, t)\}_{i=0,1,\dots,M} \quad (4.28)$$

Donde Y_i ($i = 0, 1, 2, \dots, M - 1$) es el conjunto de funciones auxiliares que reemplaza a las funciones dependientes y sus derivadas de orden inferior y f_i son las función que proveen el valor de la primera derivada de la variable auxiliar Y_i .

Así por ejemplo, para resolver la ecuación diferencial (4.24), que ya habíamos reducido como las ecuaciones (4.26), las variables auxiliares y sus derivadas serían:

$$\begin{aligned} Y_0 &= F & f_0(t, Y_0, Y_1) &= Y_1 \\ Y_1 &= G & f_1(t, Y_0, Y_1) &= -kY_0 \end{aligned} \quad (4.29)$$

Con esta identificación, el problema original puede escribirse, de forma general como la Ec. (4.28).

En este libro usaremos la rutina `odeint` de la biblioteca científica SciPy (ver *Nota* abajo) para integrar numéricamente sistemas de ecuaciones diferenciales de la forma reducida. El lector puede leer la [documentación completa de `odeint`](#)¹⁹ para conocer los detalles de su aplicación.

El primer paso para usar `odeint` es implementar las ecuaciones reducidas como una rutina. En nuestro ejemplo (Ec. 4.29) la rutina sería:

```
def ode_simple(Y, t, k=1):
    f=[0, 0]
    f[0]=Y[1]
    f[1]=-k*Y[0]
    return f
```

Los primeros dos argumentos de esta rutina (ver [Sección 4.1.3](#)), es decir Y (que contiene una lista de los valores instantáneos de las variables auxiliares Y_i) y t (el tiempo en el que las variables auxiliares tienen ese valor) deben estar, estrictamente en ese orden. Otros podrían encontrar más natural poner de primero el tiempo, pero `odeint` está diseñado para trabajar con rutinas con este *prototipo* particular. Además de estos argumentos obligatorios, la rutina puede tener cualquier otro argumento opcional. En este caso aprovechamos esta libertad para proveer el valor de la constante k , que aparece en la ecuación diferencial, y para el cual hemos asumido un valor por defecto $k=1$ (naturalmente el usuario de la rutina podrá especificar un valor distinto cuando la llame.)

Para resolver este conjunto de ecuaciones diferenciales debemos, además de la rutina anterior, proveer:

¹⁹<https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.odeint.html>

1. Valores específicos para los parámetros de la ecuación diferencial (en este caso la constante k),
2. Una lista de condiciones iniciales, es decir de los valores iniciales de las variables auxiliares $\{Y_i(t = t_0)\}$
3. un conjunto de valores del tiempo (incluyendo el tiempo inicial t_0) para los cuales deseamos predecir el valor de la antiderivada (función o funciones dependientes.)

El siguiente algoritmo prepara estos insumos para `odeint` en nuestro ejemplo particular:

```
from numpy import array

k=1.5

Y0s=array([1.0,0.0])

ts=array([0.0,1.0,2.0,3.0,4.0,5.0])
```

Nótese que para las condiciones iniciales y los valores de tiempo (que son aquí arbitrarios, el lector podría escoger unos completamente diferentes) hemos escogido usar arreglos de NumPy (`array`) en lugar de listas planas (ver [Sección 4.1.1](#)). Aunque esto no es obligatorio, más adelante hará más fácil la manipulación matemática de estas variables.

Nota

El plural en los algoritmos. Preste atención a la convención que usaremos en lo sucesivo de usar la letra `s` como sufijo del nombre de algunos arreglos y matrices (p.e. `Y0s`, `ts`). En lo sucesivo (a no ser que se indique lo contrario) `t` denotará un valor individual de la variable, pero `ts` será un arreglo de valores de `t`.

La solución numérica al conjunto de ecuaciones diferenciales implementados en la rutina `ode_simple` se obtiene, finalmente, invocando `odeint`:

```
from scipy.integrate import odeint
Ys=odeint(ode_simple,Y0s,ts,args=(k,))
```

```
Solución, Ys =
[[ 1.          0.        ]
 [ 0.33918602 -1.15214115]
 [-0.76990562 -0.78158038]
 [-0.86146852  0.6219388 ]
 [ 0.18550948  1.20348632]
 [ 0.987313   0.1944726 ]]
```

Las filas de la matriz solución `Ys`, contienen el valor de las variables auxiliares $\{Y_i\}$ en cada uno de los tiempos provistos. Las columnas, naturalmente, corres-

ponden a los valores instantáneos de cada una de esas variables auxiliares. Así, la componente $Y_s[0,0]$ corresponde al valor de Y_0 (es decir el valor de la función F de nuestro ejemplo) en t_0 (condición inicial).

También es posible extraer tajadas de la matriz. Así, $Y_s[:,1]$ (que podría leerse como *el segundo valor de cualquier fila* o simplemente *la columna 1*), corresponde al valor de la función auxiliar G en cada uno de los tiempos de integración (recuerde que $G = Y_1$, ver la identificación en la Ec. 4.29).

Usando la matriz de solución Y_s es posible, finalmente, hacer el gráfico de la función $F(t)$ que estabamos buscando:

(Algoritmo 4.3)

```
import matplotlib.pyplot as plt

#Extraemos los valores de la función F
Fs=Ys[:,0]

plt.figure();
plt.plot(ts,Fs,marker='o', linewidth=0);

#--hide--
plt.xlabel("$t$");
plt.ylabel("$F(t)$");
```

ver Figura 4.3

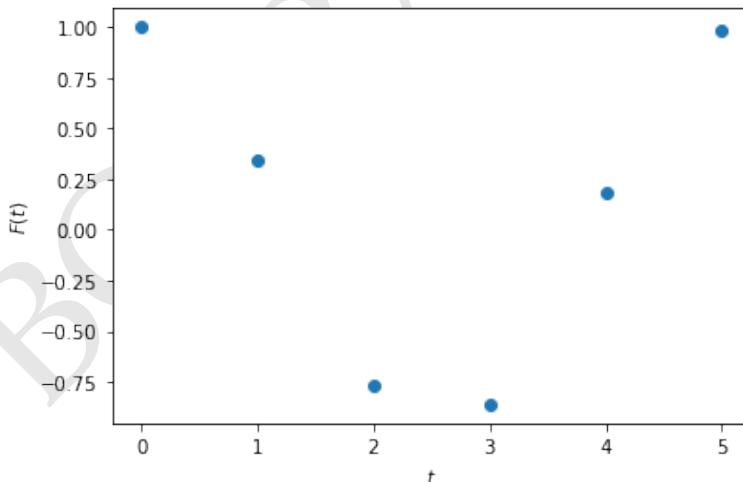


Figura 4.3: Figura correspondiente al código 4.3. Solución aproximada de la ecuación diferencial $d^2F/dt^2 = -kF$ con $k=1.5$.

Naturalmente la resolución de este gráfico es bastante pobre porque hemos pedido al algoritmo encontrar únicamente los valores de $F(t)$ en 5 valores del tiempo (arreglo ts). Si se incrementa el número de componentes de este vector el resultado será mucho más cercano al que esperamos de una función.



Nota

Los algoritmos detrás de `odeint`. La rutina `odeint` es un *empaque* en Python (*wrap* en inglés) de un complejo y robusto paquete de rutinas conocido como **ODEPACK**^a. Desarrollado por el *Center for Applied Scientific Computing* del *Lawrence Livermore National Laboratory*, las rutinas de ODEPACK están escritas en lenguaje FORTRAN77 (Python se usa únicamente para pasar los parámetros al paquete y para recuperar las salidas; ese es justamente el sentido del nombre “empaque”) y han sido probadas y perfeccionadas durante varias décadas en distintas aplicaciones científicas y de ingeniería [5].

Existen otras rutinas en el paquete SciPy para resolver ecuaciones diferenciales con condiciones iniciales (IVP). Por ejemplo `ode` y `solve_ivp` pueden usarse también (esta última es, por ejemplo, la recomendada por los desarrolladores de SciPy). Sin embargo, estas otras rutinas tienen una *interface* un poco más complicada. Así por ejemplo, para integrar la e.d.m. del ejemplo visto aquí, usando `solve_ivp`, el código **mínimo** en Python sería:

```
from scipy.integrate import solve_ivp
solucion=solve_ivp(fun=lambda t,Y:ode_simple(Y,t,k),
                    t_span=[ts[0],ts[-1]],y0=Y0,t_eval=ts)
```

Como puede apreciarse la complejidad del código supera con creces la de aquel que usamos para invocar `odeint`. A esto se suma el hecho de que la solución, que en el caso de `odeint` es una matriz `Y` fácil de interpretar, en el caso de `solve_ivp` es en realidad un *objeto* cuyo *atributo* `solucion.y` contiene la solución que buscamos. Y finalmente, pero no menos importante: para el tipo de ecuaciones diferenciales que usaremos en este libro `solve_ivp` es casi dos veces más lento que `odeint`. El lector sin embargo puede explorar esas otras alternativas, especialmente si quiere, por ejemplo, comparar distintos métodos de solución (a diferencia de `odeint`, `solve_ivp` escoger el método de solución.)

^a<https://computing.llnl.gov/casc/odepack>

4.1.10. Funcionales y cálculo de variaciones

Un tema poco cubierto en los textos básicos de cálculo, pero de gran utilidad en la mecánica, es el denominado **cálculo de variaciones**. Si bien en esta sección de “repaso” no pretendemos ofrecer una introducción detallada a esta importante área del análisis matemático, es necesario presentar aquí algunos resultados básicos que serán de utilidad para el resto del libro.

Si el cálculo infinitesimal, que repasamos en las secciones anteriores, trata sobre la variación continua de funciones de variable real, el cálculo de variaciones se ocupa de la variación de los que se conocen como **funcionales**.

En términos informales, un funcional es una “función de funciones”, es decir,

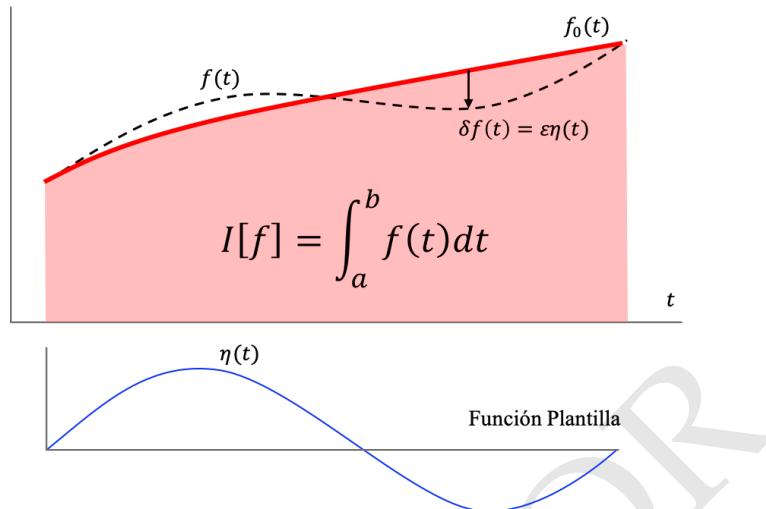


Figura 4.4: El área bajo una curva es un funcional, en tanto depende de la función que represente la curva, $f(t)$ o $f_0(t)$. Se conoce como una variación δf a la diferencia entre dos funciones cercanas, parametrizada a través de un número real ε y una función plantilla (panel inferior.) En términos de variaciones el valor de cualquier función vecina a una función de referencia f_0 se puede calcular, en un intervalo de interés, como $f(t) = f_0(t) + \varepsilon\eta(t)$.

una regla de correspondencia entre el conjunto de las funciones y el de los números reales.

Un ejemplo, muy interesante e ilustrativo de un funcional, es la integral definida de una función de variable real:

$$I[f] = \int_a^b f(t) dt$$

La notación $I[f]$, en lugar de $I(b)$ como lo usamos en [Sección 4.1.7](#), trata de poner en evidencia el hecho de que lo que nos interesa aquí no es el valor mismo de la integral definida, sino cómo el valor de esta cantidad cambia si modificamos la función f . En la [Figura 4.4](#) se muestra la interpretación gráfica de la integral definida. Sabemos que el área bajo la curva, el valor de nuestro funcional, dependerá de si usamos la función $f(t)$ o $f_0(t)$.

De la misma manera en la que se puede estudiar el efecto que un cambio muy pequeño Δt en el valor de la variable independiente t tiene en una función de variable real $f(t)$, como lo hicimos por ejemplo para definir la derivada (ver [Sección 4.1.4](#)), en el cálculo variacional es posible estudiar el efecto que un cambio pequeño δf de una función f tiene en el funcional $I[f]$.

Para hacerlo debemos primero definir otra función η que sirve de “plantilla” para el cambio. Al cambio en f se lo llama **variación** y se escribe como:

$$\delta f \equiv \varepsilon\eta \quad (4.30)$$

Una ilustración del concepto de *variación* se muestra en la [Figura 4.4](#). Allí reconocemos una importante propiedad de la función de plantilla $\eta(t)$ y es que vale

cero en los extremos del intervalo considerado $[a, b]$.

El cálculo variacional surgió originalmente para resolver problemas prácticos en física, tales como hallar las funciones que maximan o minimizan (extremos) funcionales de alguna utilidad.

Así por ejemplo, considere la siguiente pregunta: ¿cuál es la curva más corta que conecta dos puntos en el plano de euclíadiano?

Para responder a esta pregunta debemos primero construir el funcional “distancia a lo largo de una curva”, también llamado, longitud de arco ([2]):

$$I[f] = \int_a^b \sqrt{1 + \left| \frac{df}{dt} \right|^2} dt \quad (4.31)$$

Queremos encontrar la función f_0 tal que $I[f_0]$ tenga el mínimo valor entre todas las posibles funciones f .

Para encontrar la función que minimiza este funcional debemos, como se acostumbra en el cálculo ([1]), derivar el funcional respecto a la cantidad que parametriza la variación: ϵ .

Escribamos el funcional de forma más general, en términos de una función cercana al mínimo escrita como $f = f_0 + \epsilon\eta$:

$$I[f] = \int_a^b L(f(t), \dot{f}(t), t) dt \quad (4.32)$$

Nótese que hemos escrito el integrando como una función general L que depende del valor de la función $f(t)$, de su derivada $\dot{f}(t)$ y de la variable independiente t . Implícitamente, el funcional depende también del parámetro ϵ dado que $f = f_0 + \epsilon\eta$.

Si derivamos el funcional respecto de ϵ , obtenemos:

$$\frac{dI[f]}{d\epsilon} = \int_a^b \frac{d}{d\epsilon} L(f(t), \dot{f}(t), t) dt$$

Aplicando la regla de la cadena, la integral del lado derecho nos queda:

$$\frac{dI[f]}{d\epsilon} = \int_a^b \left(\frac{\partial L}{\partial f} \frac{df}{d\epsilon} + \frac{\partial L}{\partial \dot{f}} \frac{d\dot{f}}{d\epsilon} \right) dt$$

Como $f(t) = f_0(t) + \epsilon\eta(t)$, entonces $df/d\epsilon = \eta$, mientras que $d\dot{f}/d\epsilon = \dot{\eta}$. Así la integral anterior se desarrolla como:

$$\frac{dI[f]}{d\epsilon} = \int_a^b \left(\frac{\partial L}{\partial f} \eta + \frac{\partial L}{\partial \dot{f}} \dot{\eta} \right) dt \quad (4.33)$$

El término $\int_a^b (\partial L / \partial \dot{f}) \dot{\eta} dt$ se puede integrar por partes, si se hace $u = \partial L / \partial \dot{f}$ y $dv = \dot{\eta} dt$:

$$\int_a^b \frac{\partial L}{\partial \dot{f}} \dot{\eta} dt = \frac{\partial L}{\partial \dot{f}} \eta \Big|_a^b - \int_a^b \frac{d}{dx} \left(\frac{\partial L}{\partial \dot{f}} \right) \eta dt$$

El primer término del lado derecho de la ecuación anterior es cero, en tanto, por definición $\eta(a) = \eta(b) = 0$.

Reemplazando en la Ec. (4.33), la derivada del funcional respecto de epsilon queda finalmente:

$$\frac{dI[f]}{d\epsilon} = \int_a^b \eta(x) \left(\frac{\partial L}{\partial f} - \frac{d}{dx} \frac{\partial L}{\partial \dot{f}} \right) dx$$

Para que $I[f]$ sea mínima en $f = f_0$ su derivada $dI[f]/d\epsilon$ debe ser cero en $\epsilon = 0$. Esto equivale a la *ecuación integral*:

$$\int_a^b \eta(x) \left(\frac{\partial L}{\partial f} - \frac{d}{dt} \frac{\partial L}{\partial \dot{f}} \right) dt = 0 \quad (4.34)$$

que lamentablemente no es muy útil para resolver nuestro problema original. Para acercarnos a la solución necesitamos de un poderoso teorema:

Proposición 4.7

Lema fundamental del cálculo de variaciones. Si una función continua $f(t)$ en el intervalo abierto (a, b) satisface la igualdad:

$$\int_a^b f(t)h(t) dt = 0$$

para toda función $h(t)$ continuamente diferenciable (todas sus derivadas son continuas) y con *soporte compacto* (acotada), entonces $f(t) = 0$.

De acuerdo con este teorema, y suponiendo que $\eta(t)$ es continuamente diferenciable y acotada, la función entre paréntesis la ecuación integral (4.34) es:

$$\frac{\partial L}{\partial f} - \frac{d}{dt} \frac{\partial L}{\partial \dot{f}} = 0 \quad (4.35)$$

Esta ecuación es una versión particular (para funciones de una sola variable) de la que se conoce en la historia como la **ecuación de Euler-Lagrange** y que será de importancia central en este libro.

Volviendo a nuestro problema original, es decir, encontrar la curva con la menor longitud entre dos puntos, y reconociendo que:

$$L(f(t), \dot{f}(t), t) = \sqrt{1 + |\dot{f}(t)|^2},$$

Entonces $\partial L/\partial f = 0$ (no aparece el símbolo f en la fórmula de L) y $\partial L/\partial \dot{f} = \dot{f}/\sqrt{1 + |\dot{f}(t)|^2}$. De allí, la ecuación de Euler-Lagrange (4.35) en este problema se convierte en:

$$\frac{d}{dt} \left(\frac{\dot{f}}{\sqrt{1 + |\dot{f}(t)|^2}} \right) = 0$$

Esta ecuación significa que el término entre paréntesis es constante. Después de un poco de álgebra, la expresión resultante, se puede integrar para obtener:

$$f(t) = At + B,$$

donde A, B son constantes.

La respuesta final a la pregunta original es ahora clara: la curva más corta entre dos puntos en el plano euclíadiano es una línea recta.

Algoritmos en el cálculo variacional

Si el cálculo variacional es poco común en los textos básicos de cálculo infinitesimal, los algoritmos relacionados con él son aún más escasos en los textos de análisis numérico.

Dada la importancia del cálculo variacional en la mecánica nos detendremos un momento aquí para explorar desde la algoritmia, al menos la solución al problema de cálculo variacional que expusimos en la sección anterior: el cálculo de la curva más corta entre dos puntos en el plano euclíadiano.

Para ello escribamos primero la rutina que servirá en nuestro caso como funcional (y que implementa la Ec. 4.31):

(Algoritmo 4.4)

```
def funcional_integral(f0,eta,epsilon,a,b,**opciones_de_f0):

    #Definimos las función con su variación
    f=lambda t:f0(t,**opciones_de_f0)+epsilon*eta(t)

    #La derivada de f la calculamos con derivative
    from scipy.misc import derivative
    dfdt=lambda t:derivative(f,t,0.01)

    #Este es el integrando del funcional
    from numpy import sqrt
    L=lambda t:sqrt(1+abs(dfdt(t))**2)

    #El funcional es la integral definida del integrando
    from scipy.integrate import quad
    integral=quad(L,a,b)
    longitud=integral[0]

    return longitud
```

Nótese que un *funcional* en el lenguaje de la algoritmia es una rutina que recibe como parámetros otras rutina (en este caso f_0 y η) y devuelve un valor numérico (en este caso $longitud$.)

La rutina en el Alg. (4.4), si bien parece compleja, recoje todos los elementos que hemos aprendido en esta sección: los parámetros opcionales de una rutina expresados como `**opciones_de_f0` y que vimos en una nota de la Sección 4.1.3, las funciones `lambda` que vimos en la misma sección, la derivada numérica calculada usando `derivative` que conocimos en la Sección 4.1.4 y la integral por cuadraturas usando `quad` de la Sección 4.1.7.

Más importante aún es el hecho que esta rutina puede usarse para cualquier

funcional que se exprese como una integral definida de la forma de la Ec. (4.32). Para adaptarla a otras situaciones, simplemente se debe cambiar la función L. En la sección de problemas al final de este capítulo se pone a prueba esta rutina en otros contextos.

Supongamos ahora que queremos calcular la curva más corta que une los puntos del plano cartesiano $(0,0)$ y $(\pi, 1)$ (es decir $a = 0$ y $b = \pi$). Para ello proponemos una función de referencia $f_0(t) = (t/\pi)^n$. Esta función pasa por ambos puntos para todo n . Como función de plantilla $\eta(t)$, que debe ser una función acotada de acuerdo al lema fundamental del cálculo de variaciones, usaremos la función trigonométrica seno (que cumple la condición $\eta(a) = \sin 0 = 0$ y $\eta(b) = \sin \pi = 0$).

El siguiente algoritmo implementa estas elecciones:

```
#Intervalo entre los puntos
from numpy import pi
a=0
b=pi

#Funcion de referencia
def curva(t,n=1):
    return (t/pi)**n

#Función plantilla
from numpy import sin
eta=sin
```

Para ilustrar el uso de la rutina en el Alg. (4.4), calculemos la longitud de arco para el caso en el que $n = 2$ y $\epsilon = 0,5$:

```
n=2
If=funcional_integral(curva,eta,0.5,a,b,n=n)

I[f] = 3.337162809417341
```

Para encontrar la trayectoria más corta entre los puntos seleccionados, debemos minimizar una función del tipo `longitud_arco(epsilon)` que llame a la rutina `funcional_integral`, pero que solo dependa de la variable que queremos minimizar, es decir de `epsilon`. Para ello podemos definir la función `lambda`:

```
longitud_arco=lambda epsilon:funcional_integral(curva,eta,epsilon,
                                                a,b,n=n)
```

La minimización, finalmente, se consigue usando la rutina `minimize` del paquete SciPy, capaz de encontrar el mínimo de funciones escalares con un número arbitrario de variables. Lo único que necesita `minimize` para lograr su cometido es que le pasemos una rutina que tenga un solo parámetro, en nuestro caso `longitud_arco` y un valor de prueba para la variable independiente (en nuestro caso usaremos $\epsilon = 0$):

(Algoritmo 4.5)

```
from scipy.optimize import minimize
solucion=minimize(longitud_arco,0.0)
```

Resultado de la minimización:

```
fun: 3.2975722013512403
hess_inv: array([[0.73687233]])
jac: array([1.1920929e-06])
message: 'Optimization terminated successfully.'
nfev: 12
nit: 3
njev: 4
status: 0
success: True
x: array([0.25801323])
```

Nótese que el resultado de la rutina `minimize` es un *objeto* entre cuyos atributos se encuentra el valor de la variable independiente `x` que hace mínima la función de nuestro interés, en este caso `longitud_de_arco`.

Puesto en términos de nuestro problema el resultado anterior indica que para curvas del tipo $f_0(t) = (t/\pi)^2$, que sufren variaciones con una función plantilla $\eta(t) = \sin t$, la curva de mínima longitud entre el punto $(0,0)$ y el punto $(0,\pi)$, corresponde a una variación con $\epsilon = 0,258$.

Hagamos un gráfico de la función resultante y de su comparación con la solución analítica:

(Algoritmo 4.6)

```
import matplotlib.pyplot as plt
plt.figure()

from numpy import linspace,pi
ts=linspace(0,pi)

#Valor de epsilon proveniente de la minimización
epsilon=solucion.x[0]

plt.plot(ts,curva(ts,n=n), 'r.',
         label=f"Curva de referencia")
plt.plot(ts,curva(ts,n=n)+epsilon*eta(ts), 'b-',
         label=f"Curva variada con $\epsilon$={epsilon:g}")
plt.plot(ts,curva(ts,n=1), 'k--',
         label=f"Línea recta")

plt.legend();

---hide---
plt.xlabel("t");
```

ver Figura 4.5

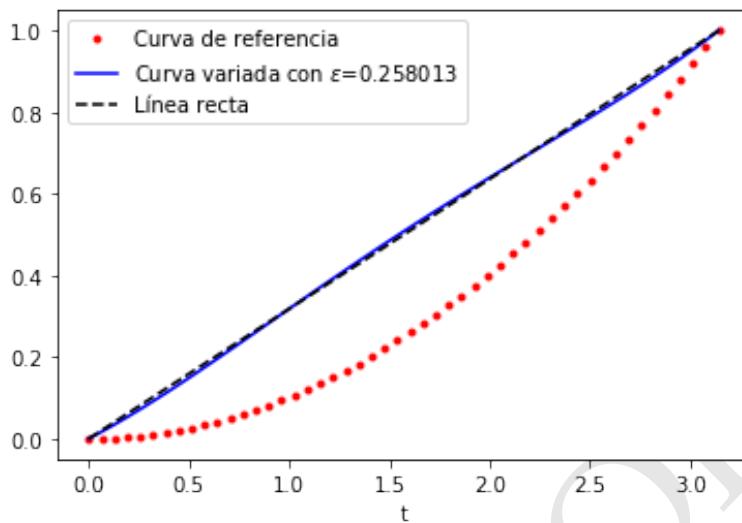


Figura 4.5: Figura correspondiente al código 4.6. La curva continua indica una aproximación numérica al camino más corto entre los puntos $(0, 0)$ y $(0, \pi)$ del plano euclíadiano, encontrada al minimizar el funcional longitud de arco y usando como función de prueba $f_0 = (t / \pi)^n$ (línea punteada) y como función plantilla $\epsilon(t) = \sin t$. El valor de ϵ que corresponde a la solución se muestra en la etiqueta. Para comparación se muestra (línea rayada) la solución exacta, que corresponde a una línea recta.

4.1.11. Gráficos interactivos

Para ver los gráficos interactivos use a las libretas de Jupyter que están disponibles en la versión electrónica del libro.

4.1.12. Series infinitas

En las secciones anteriores hemos considerado las funciones como entidades abstractas. En todos los casos asumimos que el valor de esas funciones puede obtenerse con algún procedimiento que no hemos especificado. Considere por ejemplo el caso de la función trigonométrica seno. Sabemos como se define, conocemos su relación con otras funciones, su antiderivada y todas las derivadas de orden arbitrario de la función. Pero ¿sabemos cómo calcular el valor de $\sin t$?

Para ser estrictos las únicas funciones cuyos valores se pueden calcularse de manera elemental, es decir ejecutando un número finito de sumas (o restas), multiplicaciones y divisiones son las funciones polinómicas y las funciones racionales.

Definimos una función polinómica como aquella que puede escribirse de la forma:

$$P_k(t) = a_0 + a_1 t + a_2 t^2 + \dots + a_k t^k$$

decimos que $P_k(t)$ es un polinomio de orden k .

Las funciones racionales son aquellas que pueden escribirse de la forma:

$$R(t) = \frac{P_k(t)}{Q_m(t)}$$

done $P_k(t)$ y $Q_l(t)$ son polinomios de orden k y m respectivamente.

Series de potencias

El primer resultado de interés para el cálculo del valor de funciones no polinómicas ni racionales depende del siguiente teorema:

Proposición 4.8

Teorema de Taylor. Si una función $f(t)$ tiene derivadas hasta orden k en el punto $t = t_0$ entonces el polinomio:

$$P_k(x) = \sum_{n=0}^k \frac{f^{(n)}(t_0)}{n!} (t - t_0)^n$$

ofrece una aproximación al valor de la función para valores de $t \rightarrow t_0$. En otras palabras:

$$\lim_{t \rightarrow t_0} [f(t) - P_k(t)] = 0$$

En términos prácticos, el teorema de Taylor se puede escribir diciendo que para valores de t cercano a t_0 , la función $f(t)$ se puede escribir como:

$$\text{Para } t \rightarrow t_0 : f(t) \approx \sum_{n=0}^k \frac{f^{(n)}(t_0)}{n!} (t - t_0)^n$$

A la suma de la forma del lado izquierdo de la ecuación anterior, $s_k \equiv \sum_{n=0}^k a_n$ la llamamos en general una **suma parcial** y a la secuencia $\{s_0, s_1, s_2, \dots\}$ se la conoce como una **serie infinita** o en breve una **serie**.

En particular a la serie formada por las sumas parciales en el teorema de Taylor la llamamos una **serie de potencias** o **serie de Taylor**.

Definición 4.2

Serie convergente. Decimos que una serie $\{s_0, s_1, s_2, \dots\}$ converge a S si:

$$\lim_{k \rightarrow \infty} s_k = S$$

en otros términos:

$$\sum_{n=0}^{\infty} a_n \equiv \lim_{k \rightarrow \infty} \sum_{n=0}^k a_n = S$$

En la definición anterior hemos introducido la notación $\sum_{n=0}^{\infty} a_n$ para representar el valor al que converge una serie (si lo hace.) Esta notación es corriente en el cálculo y será muy utilizada en este libro.

En términos de esta notación y del concepto de convergencia, el teorema de Taylor se puede escribir como:

$$f(t) = \lim_{t \rightarrow t_0} \sum_{n=0}^{\infty} \frac{f^{(n)}(t_0)}{n!} (t - t_0)^n \quad (4.36)$$

Ahora bien, en algunos casos la serie de potencias asociada a una función puede converger incluso en puntos a una distancia finita de t_0 . Consideré por ejemplo la serie de potencias de la función exponencial $\exp(t)$ alrededor de $t_0 = 0$ (ver problemas al final del capítulo):

$$\exp(t) \approx s_k = \sum_{n=0}^k \frac{t^n}{n!}$$

Es posible mostrar que esta serie converge para cualquier valor real de t . Por la misma razón, usando la notación en la Def. (4.2) podemos escribir:

$$\exp(t) = \sum_{n=0}^{\infty} \frac{t^n}{n!} \quad (4.37)$$

e introducir una nueva definición:

Definición 4.3

Convergencia uniforme y función analítica. Decimos que una serie s_k , cuyos coeficientes son función de una variable independiente t , converge *uniformemente* a $f(t)$ si lo hace para todo valor de t que cumple $|t - t_0| < r$, donde $r > 0$ se conoce como el *radio de convergencia*. En otras palabras:

$$f(t) = \sum_{n=0}^{\infty} a_n(t)$$

A la función $f(t)$ que cumple esta condición la llamamos una **función analítica**.

Muchas de las series de potencias definidas con la Ec. (4.36) convergen uniformemente al valor de la función que expanden. En ese caso escribimos:

$$f(t) = \sum_{n=0}^{\infty} \frac{f^{(n)}(t_0)}{n!} (t - t_0)^n \quad (4.38)$$

Las series de taylor de algunas funciones analíticas conocidas que serán de gran utilidad en la mecánica celeste, así como su radio de convergencia uniforme, son provistas en la lista a continuación:

- **Funciones seno y coseno.** Para todo $t \in \mathbb{R}$:

$$\sin t = \sum_{n=0}^{\infty} \frac{(-1)^n t^{2n+1}}{(2n+1)!} = t - \frac{t^3}{3!} + \frac{t^5}{5!} - \frac{t^7}{7!} + \dots \quad (4.39)$$

$$\cos t = \sum_{n=0}^{\infty} \frac{(-1)^n t^{2n}}{2n!} = 1 - t^2 + \frac{t^4}{4!} - \frac{t^6}{6!} + \dots \quad (4.40)$$

■ **Funciones seno y coseno hiperbólicas.** Para todo $t \in \mathbb{R}$:

$$\sinh t = \sum_{n=1}^{\infty} \frac{t^{2n+1}}{(2n+1)!} = t + \frac{t^3}{3!} + \frac{t^5}{5!} + \frac{t^7}{7!} + \dots \quad (4.41)$$

$$\cosh t = \sum_{n=0}^{\infty} \frac{t^{2n}}{2n!} = 1 + t^2 + \frac{t^4}{4!} + \frac{t^6}{6!} + \dots \quad (4.42)$$

Cálculo numérico de series infinitas

Las series de potencias juegan un papel central en la computación porque permiten reducir el cálculo de funciones que pueden ser muy complicadas a la ejecución de un conjunto finito de operaciones numéricas simples: sumas (o restas), multiplicaciones y divisiones. Sin embargo usar un computador para calcular eficientemente el valor de convergencia de una serie infinita no es una tarea trivial.

Tres problemas comunes se enfrentan al calcular series infinitas en un computador. Para empezar la *precisión* con la que podemos representar cantidades numéricas en un computador está limitada a un cierto número de dígitos significativos. Para ilustrarlo, considere esta operación:

`x=1+1e-16`

`1.0000000000000000000000000`

Llamamos a la cantidad *epsilon* ($\approx 10^{-16}$) que cumple la condición `1+epsilon=1` la *precisión de la máquina*. Esta limitación implica que para un número representado en notación científica:

$$\text{mantisa} \times 10^{\text{exponente}}$$

donde $0 \leq \text{mantisa} < 1$, tan solo los primeros 16 dígitos decimales son realmente *significativos*. Para hacer esto más evidente considere esta operación:

`x=0.123456789012345678901234567890`

`0.123456789012345677369886232100`

Nótese que aunque hemos querido almacenar 30 dígitos significativos, tan solo los primeros 16 fueron efectivamente recuperados de la memoria. A partir del decimal 17 lo que obtenemos son cifras al azar.

Al manipular algebraicamente cantidades que tienen esta limitación, los resultados pueden ser aún menos significativas. En un algoritmo (en especial aquellos con los que se calculan el valor de convergencia de series) después de muchas operaciones entre cantidades con precisión limitada el resultado puede llegar a acumular importantes *errores de redondeo*.

Para ilustrar la manera como los errores de redondeo se propagan considere el cálculo de esta sencilla cantidad:

$$(Nx + y) - Nx$$

donde $N \gg 1$. Si bien matemáticamente el resultado es obviamente y , en el computador obtenemos:

```
N=1e12
x=1.0
y=0.1
z=(N*x+y)-(N*x)
```

```
x = +1.000000000000000e+00
y = +1.000000000000000e-01
z = +9.997558593750000e-02
```

A este tipo de errores, que se producen normalmente cuando se restan o dividen cantidades muy parecidas, se los conoce como errores de cancelación subtractiva y pueden llegar a ser muy importantes, por ejemplo, al calcular series alternantes.

Para ilustrar como los errores de cancelación subtractiva pueden volverse significativos en la estimación del valor de convergencia de una serie alternante consideren el siguiente algoritmo en el que se calcula la suma parcial de la serie de Taylor para la función seno:

```
from math import factorial
from numpy import pi

#Número de términos de la serie
k=18
#Valor del ángulo
t=3*pi
#Suma parcial
sk=0
for n in range(k):
    sk+=(-1)**n*t**(2*n+1)/factorial(2*n+1)
```

```
s_18(9.42478) = -7.6538436107548528e-08
sin(9.42478) = 3.6739403974420594e-16
```

Si bien el resultado obtenido con la suma parcial es correcto hasta la séptima cifra significativa ($\sim 10^{-7}$), hay una manera más eficiente de calcular este resultado evitando la propagación de errores de redondeo (cancelación subtractiva). Podemos aprovechar la propiedad de la función:

$$\sin(t) = \sin(t + 2n\pi)$$

para que reconocer que $\sin(3\pi) = \sin(\pi)$ y calcular la suma parcial en el valor más pequeño de t :

```
#Valor de t mapeado en el intervalo [0,2 pi)
from numpy import mod
tn=mod(t,2*pi)

#Suma parcial
sk=0
for n in range(k):
    sk+=(-1)**n*mod(tn,2*pi)**(2*n+1)/factorial(2*n+1)

s_18(9.42478) = 3.3280566951896521e-16
sin(9.42478) = 3.6739403974420594e-16
```

La diferencia en los dos resultados se hace notar inmediatamente.

Otro truco que puede ser útil, se ilustra en el cálculo de la función:

$$\exp(-t) = \sum_{n=0}^{\infty} \frac{(-1)^n t^n}{n!} \quad (4.43)$$

(Algoritmo 4.7)

```
#Número de términos en la serie
k=10
#Valor de t
t=5.0
#Suma parcial de exp(-t)
sk=0
for n in range(k):
    sk+=(-1)**n*t**n/factorial(n)

s_10(1) = -1.8271053791887111e+00
exp(1) = 2.7182818284590451e+00
```

Si en lugar de la serie alterna usamos la identidad $\exp(-t) = 1 / \exp(t)$ y calculamos la suma parcial correspondiente a $\exp(t)$ (Ec. 4.37) obtenemos:

(Algoritmo 4.8)

```
#Suma parcial de exp(t)
sk=0
for n in range(k):
    sk+=t**n/factorial(n)

1/s_10(5) = 6.9594528636495370e-03
exp(5) = 6.7379469990854670e-03
```

El tercer problema es la *convergencia lenta*, que puede hacer que se requieran decenas o cientos de términos de una serie para conseguir una precisión aceptable en el valor estimado de convergencia. La convergencia lenta no solo aumenta el tiempo requerido para calcular una serie, sino que también puede hacer que los errores de redondeo y cancelación sustractiva se acumulen al punto en el que el resultado sea más error que otra cosa.

La siguiente rutina implementa el cálculo de la serie de potencias de la función exponencial pero en lugar de escoger un número fijo de términos de la suma

parcial, como lo hicimos en los Algs. (4.7 y (4.8), la rutina va agregando nuevos términos hasta que consigue un nivel de precisión o *tolerancia* deseado $\delta \ll 1$. Para esto, por cada término que agrega a la serie calcula el error relativo el valor de la suma como:

$$\Delta_n \equiv \left| \frac{s_{n+1} - s_n}{\bar{s}} \right|$$

donde $\bar{s} = (s_n + s_{n+1})/2$ es el promedio entre dos sumas parciales sucesivas.

El cálculo de la serie converge cuando se cumple:

$$\Delta_n < \delta$$

(Algoritmo 4.9)

```
def exp_taylor(t,delta=1e-15):
    #Valor inicial del error relativo (número arbitrario)
    Dn=1
    #Primer valor de la suma parcial
    sn=0
    #Inicializa contador de términos
    n=0
    #El ciclo continúa mientras no se alcance la tolerancia
    while Dn>delta:
        #Almacena el último valor de la suma
        s=sn
        #Calcula el término n-esimo de la suma
        from math import factorial
        an=t**n/factorial(n)
        #Agrega el término a la suma n-esima
        sn+=an
        #Promedio de dos sumas parciales sucesivas
        smed=(sn+s)/2
        #Error relativo
        Dn=abs(an/smed)
        #Incremento en el contador n
        n+=1
    return sn,Dn,n
```

Aunque esta no es la forma más compacta y efectiva de calcular la función exponencial, la mayoría de las rutinas que usan iteraciones en este libro tendrán una estructura y notación similares a esta. Además, esta forma de escribirla nos permitirá ilustrar algunos de las limitaciones mencionados anteriormente.

Considere por ejemplo el cálculo de la función $\exp(-t)$:

```
t=20
x,error_x,n_x=exp_taylor(-t)
y,error_y,n_y=exp_taylor(+t)
```

```
x = exp(-20) = 5.4781029165292094e-10 (error 7.00e-16, n = 96)
1/y = 1/exp(+20) = 2.0611536224385591e-09 (error 9.22e-16, n = 66)
exp(-20) = 2.0611536224385579e-09
```

Nótese que todos los efectos descritos hasta aquí son otra vez evidentes: (1) el error en la serie alternada es grande y debido principalmente a cancelaciones substractivas, (2) la serie alterna requiere un número mayor de términos para conseguir el mismo nivel de precisión de la serie no alterna (convergencia lenta.)

Otras series

A continuación se enumeran otras series que serán de utilidad en el texto:

- **Serie geométrica:** para $t \neq 1$,

$$\frac{1}{1-t} = \sum_{n=0}^{\infty} t^n = 1 + t + t^2 + t^3 + \dots \quad (4.44)$$

- **Serie binomial:** para todos los valores de $a \in \mathbb{R}$ y $|t| < 1$,

$$(1+t)^a = \sum_{n=0}^{\infty} \binom{a}{k} t^n = 1 + at + \frac{a(a-1)}{2!} t^2 + \frac{a(a-1)(a-2)}{3!} t^3 + \dots \quad (4.45)$$

Los coeficientes de esta serie tienen la forma general:

$$\binom{a}{k} \equiv \frac{a(a-1)(a-2) \cdots (a-k+1)}{k!}$$

Cuando para aproximar la función $(1+t)^a$ se usa una de las sumas parciales de la serie binomial, decimos que se hace una **aproximación binomial de $(1+t)^a$** .

Series de Stumpff

Una familia de funciones analíticas de gran interés en la mecánica celeste son las funciones o **series de Stumpff**:

$$c_k(t) = \sum_{n=0}^{\infty} \frac{(-1)^n t^n}{(2n+k)!} = \frac{1}{k!} - \frac{t}{(k+2)!} + \frac{t^2}{(k+4)!} - \frac{t^3}{(k+6)!} + \dots$$

que convergen para todo $t \in \mathbb{R}$

Estas series tienen una serie de propiedades muy interesantes que serán explotadas en el ?? y que enumeramos a continuación:

- **Relación de recurrencia:** Si se conoce el valor de la función de orden $k+2$ es posible calcular el valor de la función k usando:

$$c_k(t) = \frac{1}{k!} - tc_{k+2}(t) \quad (4.46)$$

- **Relación de recurrencia para las derivadas:** Las derivadas de las series de Stumpff satisfacen las siguientes relaciones de recurrencia:

$$\begin{aligned} \frac{d}{dt} [c_0(\alpha s^2)] &= -\alpha s c_1(\alpha s^2) \\ \frac{d}{dt} [t^{k+1} c_{k+1}(\alpha s^2)] &= s^k c_k(\alpha s^2) \quad \text{para } k = 0, 1, 2, \dots \end{aligned} \quad (4.47)$$

donde α es cualquier número real. La parametrización de la variable independiente de las series de Stumpff en la forma αs^2 será muy común en mecánica celeste.

- **Equivalencia con las funciones trigonométricas e hiperbólicas:** Todas series de Stumpff se pueden escribir en términos de las funciones trigonométricas e hiperbólicas. Por esta razón se consideran una generalización de estas últimas. Se puede probar, usando las series de Taylor en las Ecs. (4.39)-(4.40) que las primeras dos series de Stumpff se pueden escribir como:

$$c_0(\alpha s^2) = \begin{cases} \cos(\alpha^{1/2}s) & \text{Si } \alpha \geq 0 \\ \cosh(|\alpha|^{1/2}s) & \text{Si } \alpha < 0 \end{cases} \quad (4.48)$$

y

$$c_1(\alpha s^2) = \begin{cases} \sin(\alpha^{1/2}s)/(\alpha^{1/2}s) & \text{Si } \alpha \geq 0 \\ \sinh(|\alpha|^{1/2}s)/(|\alpha|^{1/2}s) & \text{Si } \alpha < 0 \end{cases} \quad (4.49)$$

- **Relaciones de escalado:** Si bien las series de Stumpff convergen para todos los valores reales de la variable independiente, el cálculo numérico de las mismas para valores muy grandes de $|t|$ puede sufrir de errores de redondeo o ser muy ineficiente. Como vimos en el caso del cálculo numérico de la serie de Taylor del seno, una solución posible para este inconveniente es la de mapear un valor arbitrario de t en un intervalo más pequeño. En el caso de la función coseno, por ejemplo, se puede calcular el coseno de cualquier ángulo grande, por ejemplo $2t$ usando del coseno del ángulo más pequeño t y la identidad:

$$\cos(2t) = 2\cos(t)^2 - 1$$

Las siguientes juegan un papel análogo en el caso del cálculo de las funciones de Stumpff.

$$c_0(4t) = 2c_0(t)^2 - 1 \quad (4.50)$$

$$c_1(4t) = c_0(t)c_1(t) \quad (4.51)$$

$$c_2(4t) = \frac{1}{2}c_1(t)^2 \quad (4.52)$$

$$c_3(4t) = \frac{1}{4}c_2(t) + \frac{1}{4}c_0(t)c_3(t) \quad (4.53)$$

$$(4.54)$$

- **Regla iterativa:** Las series de Stumpff se pueden calcular también usando la regla iterativa (ver problemas al final del capítulo):

$$p_n = \frac{t}{(2n+k+1)(2n+k+2)} (1 - p_{n+1}) \quad (4.55)$$

Si se comienza con un valor $p_N = 0$ para un valor de N suficientemente grande, al descender con la regla iterativa hasta p_0 se obtiene una aproximación para la serie de Stumpff de orden k :

$$c_k(t) \approx \frac{1}{k!} (1 - p_0)$$

El error de esta aproximación es aproximadamente igual a:

$$\epsilon_N = \frac{t^N}{(2N+k+2)!} \quad (4.56)$$

En el ?? hemos incluido una corta rutina, `serie_stumpff(t, k)` que usa la regla iterativa para calcular el valor de las series de Stumpff de forma muy eficiente. Para ilustrar el uso de esta rutina, en el algoritmo a continuación ponemos a prueba algunas de las propiedades mencionadas en esta sección.

Así por ejemplo, podemos verificar la relación de recurrencia (Ec. 4.46):

```
#Verifica las reglas de recurrencia
t=2
k=4

from pymcel.export import serie_stumpff
ck=serie_stumpff(t,k)
ckp2=serie_stumpff(t,k+2)
```

Regla de recurrencia:
 $c_4(2) = 0.03898592369134362$
 $1/4! - t c_6(2) = 0.03898592369134361$

```
#Verifica las reglas de escalado
t=2
k=4

from pymcel.export import serie_stumpff
c0_4t=serie_stumpff(4*t,0)
c0_t=serie_stumpff(t,0)
c1_4t=serie_stumpff(4*t,1)
c1_t=serie_stumpff(t,1)
c2_4t=serie_stumpff(4*t,2)
c2_t=serie_stumpff(t,2)
c3_4t=serie_stumpff(4*t,3)
c3_t=serie_stumpff(t,3)
```

```
c_0(8) = -0.9513631281258474, con escalado = -0.9513631281258474
c_1(8) = 0.10891980905843202, con escalado = 0.10891980905843208
c_2(8) = 0.24392039101573093, con escalado = 0.24392039101573093
c_3(8) = 0.111385023867696, con escalado = 0.11138502386769598
```

Series de Fourier

A lo largo de esta sección hemos visto como es posible aproximar algunas funciones analíticas usando series de potencias o series de polinomios. Pero los polinomios no son las únicas funciones capaces de realizar esta tarea. Así por ejemplo, cualquier familia contable de funciones de variable real $f_n(t)$, que en un determinado dominio, obedezca la condición:

$$\int_{\mathcal{D}} g_n(t)g_m(t) dt = 0, \text{ Si } n \neq m$$

puede aproximar funciones definidas en el dominio \mathcal{D} con series del tipo:

$$f(t) \approx \sum_{n=1}^k A_n g_n(t)$$

donde:

$$A_n = \frac{\int_{\mathcal{D}} g_n(t)f(t) dt}{\int_{\mathcal{D}} g_n(t)^2 dt} \quad (4.57)$$

Uno de los conjuntos contables de funciones más notables usados en la historia para aproximar otras funciones es el de las funciones trigonométricas seno y coseno definidas en el dominio finito $\mathcal{D} = [0, T]$:

$$\begin{aligned} g_0(t) &= 1/2 \\ g_{2n-1}(t) &= \sin(n\omega t) \\ g_{2n}(t) &= \cos(n\omega t) \end{aligned}$$

donde $n = 1, 2, \dots$ y $\omega \equiv 2\pi/T$. Es posible mostrar que:

$$\begin{aligned} \int_0^T g_n(t)g_m(t) dt &= 0 \\ \int_0^T g_n(t)^2 dt &= \frac{T}{2} \end{aligned} \quad (4.58)$$

Las series resultantes, que se puede escribir como

$$s_k(t) = \frac{1}{2}A_0 + \sum_{n=1}^k A_n \cos(n\omega t) + \sum_{n=1}^k B_n \sin(n\omega t) \quad (4.59)$$

y que permiten aproximar funciones $f(t)$ en el dominio $[0, T]$ se conocen como **Series de Fourier**.

Si una función $f(t)$ es aproximada con una serie de Fourier, los coeficientes correspondientes se calculan usando la Ec. (??):

$$A_0 = \frac{2}{T} \int_0^T f(t) dt \quad (4.60)$$

$$A_n = \frac{2}{T} \int_0^T \cos(n\omega t) f(t) dt \quad (4.61)$$

$$B_n = \frac{2}{T} \int_0^T \sin(n\omega t) f(t) dt \quad (4.62)$$

(4.63)

Como sucede con las series de potencias, una cosa es aproximar una función en un dominio usando series de funciones y otra muy distinta es asegurar la convergencia uniforme de la serie infinita. En el caso de las series de Fourier el teorema de Dirichlet ofrece las condiciones para dicha convergencia:

Proposición 4.9

Teorema de Dirichlet para series de Fourier. Dada una función $f(t)$ definida en el dominio $[0, T]$ y que cumple las siguientes condiciones dentro de este intervalo:

1. Es integrable.
2. Esta acotada.
3. Tiene un número finito de discontinuidades.

La serie de Fourier (Ec. 4.59) converge uniformemente a $f(t)$ para cualquier valor de $t \in [0, T]$

Para ilustrar lo visto aquí sobre las series de Fourier, a continuación presentamos el algoritmo de una rutina que calcula el valor de los coeficientes A_0 , A_n y B_n para cualquier función provista por el usuario. El algoritmo se vale de la rutina `quad` para calcular numéricamente las integrales en las Ecs. (4.60)-(4.62):

(Algoritmo 4.10)

```
def coeficientes_fourier(funcion,T,k,args=()):
    #Funciones externas
    from scipy.integrate import quad
    from numpy import sin,cos

    #Parametro omega
    w=2*pi/T

    #Determina los coeficientes en t:
    f=lambda t:funcion(t,*args)
    As=[2*quad(f,0,T,args=args)[0]/T]
    Bs=[0]
    for n in range(1,k+1):
        f_cos_n=lambda t:funcion(t,*args)*cos(n*w*t)
        As.append(2*quad(f_cos_n,0,T,args=args)[0]/T)
        Bs.append(2*quad(f_cos_n,0,T,args=args)[0]/T)

    return As,Bs
```

```

As+=[2*quad(f_cos_n,0,T)[0]/T]
f_sin_n=lambda t:funcion(t,*args)*sin(n*w*t)
Bs+=[2*quad(f_sin_n,0,T)[0]/T]

return As,Bs

```

Nótese que el usuario debe proveer la lista de las opciones de la función `funcion` en la forma de la *tupla args* (ver Sección 4.1.3). Esta rutina solo permite calcular el valor de los coeficientes de la serie (almacenados en las listas `As` y `Bs`). Para calcular la suma parcial se debe usar la siguiente rutina:

```

def serie_fourier(t,As,Bs,T,k):
    from numpy import sin,cos

    #Parametro omega
    w=2*pi/T

    #Calcula la suma
    sk=As[0]/2
    for n in range(1,k+1):
        sk+=As[n]*cos(n*w*t)+Bs[n]*sin(n*w*t)

    return sk

```

Como un ejemplo, calculemos los primeros 30 coeficientes de la serie de Fourier de una línea recta:

```

def f(t):
    y=t
    return y
#Dominio de aproximación
T=2*pi
#Número de coeficientes
k=30
#Coeficientes
As,Bs=coeficientes_fourier(f,T,k)

As = [ 6.283  0.   -0.   ... -0.   -0.   0.   ]
Bs = [ 0.   -2.   -1.   ... -0.071 -0.069 -0.067]

```

Para comprobar el resultado podemos hacer un gráfico de la rutina `serie_fourier` con algunos o todos los coeficientes calculados:

(Algoritmo 4.11)

```

#Valores de la variable independiente
import matplotlib.pyplot as plt
fig=plt.figure();
ax=fig.gca();

#Aproximaciones
from numpy import linspace

```

```

ts=linspace(0,T,200)
k=1
ax.plot(ts,serie_fourier(ts,As,Bs,T,k),label=f" k = {k}");
k=5
ax.plot(ts,serie_fourier(ts,As,Bs,T,k),label=f" k = {k}");
k=10
ax.plot(ts,serie_fourier(ts,As,Bs,T,k),label=f" k = {k}");
#Gráfico de la función
ts=linspace(0,T,30)
ax.plot(ts,f(ts), 'k+',label="Función");

#Decoración
ax.legend();
ax.set_xlabel("$t$");
ax.set_ylabel("$f(t)$");

```

ver Figura 4.6

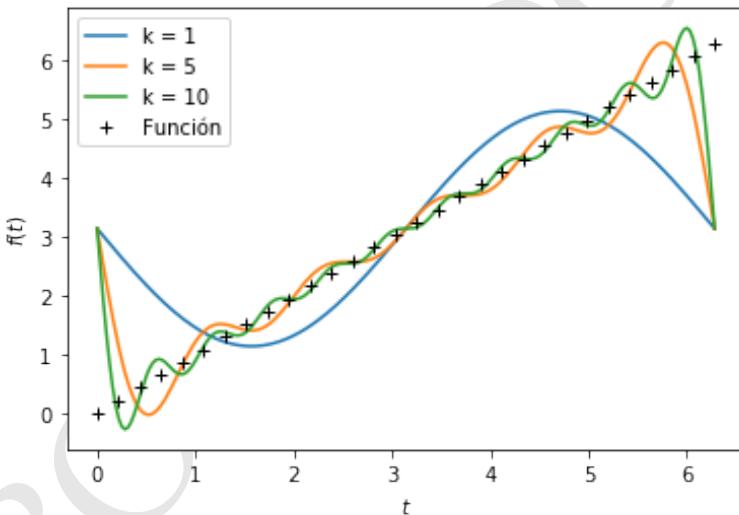


Figura 4.6: Figura correspondiente al código 4.11.

En las libretas provistas con la versión electrónica del libro²⁰ encontrarán una versión interactiva de este gráfico.

4.2. Curvas cónicas

En el año 1609, Johannes Kepler descubrió uno de los secretos mejor guardados del Universo: el camino que seguía el planeta Marte alrededor del Sol no era un círculo, como lo “mandaban” siglos de tradición filosófica y astronómica, sino una *elipse*.

²⁰<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

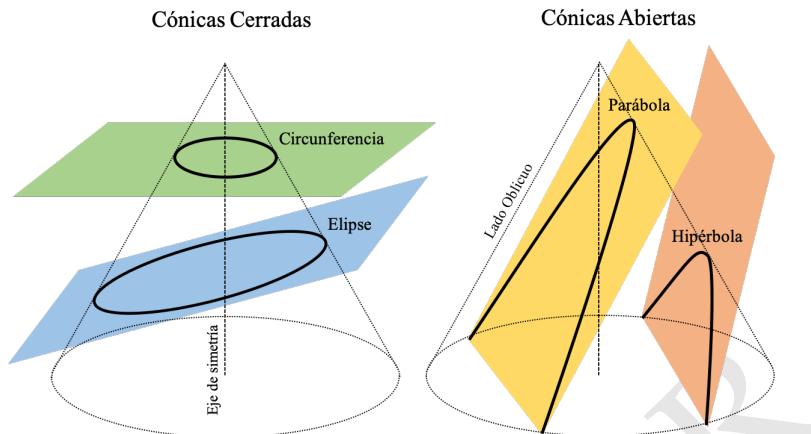


Figura 4.7: Definición geométrica original de las curvas cónicas.

Durante meses el astrónomo Prusiano había estado luchando, sin mucha suerte, por ajustar las precisas observaciones del astrónomo Danés Tycho Brahe (“[tico braja](#)”²¹) del planeta en cuestión, al modelo que Nicolás Copernico había desarrollado unos 60 años antes y en el que se suponía que los planetas se movían “alrededor” del Sol sobre trayectorias circulares descentradas (el Sol no ocupaba realmente el centro en el sistema Copernicano.)

Después de muchos intentos fallidos Kepler relata, en la que hoy se considera su obra cumbre “*Astronomía Nueva*”, que desesperado empezó a considerar la posibilidad de que la órbita de Marte fuera “ovalada” (con forma de huevo) en lugar de circular. Finalmente, después de muchos intentos, Kepler “adivinó” que el ovalo no podía ser otra cosa sino una elipse, una figura geométrica que había sido ampliamente estudiada por los geómetras de la antigüedad y la edad media, pero cuyo papel en la astronomía no había sido considerado hasta ese momento.

Esta historia marcó el inicio de la mecánica celeste y el renacimiento del interés astronómico por la elipse y las curvas emparentadas con ella y que hoy llamamos *curvas cónicas*. En las próximas sesiones repasaremos las propiedades geométricas de las cónicas, desde su definición original hasta su descripción algebraica moderna, en preparación para su aplicación en el estudio de la trayectoria de los cuerpos en mecánica celeste.

4.2.1. Definición geométrica

Desde los primeros trabajos geométricos griegos, compilados y organizados por Euclides de Alejandría (323 a 283 a.e.c.) en su libro “*Elementos*”, se sabe que la familia de curvas que resultan de intersectar un plano con un cono (una figura que se forma al hacer rotar en el espacio un triángulo alrededor de uno de sus lados, ver ??) tienen propiedades geométricas especiales. Es a esta familia de curvas a las que llamamos *cónicas*, en clara referencia a su definición geométrica original.

La **circunferencia**, que es una cónica, resulta por ejemplo de intersectar el cono

²¹https://es.forvo.com/word/tycho_brahe

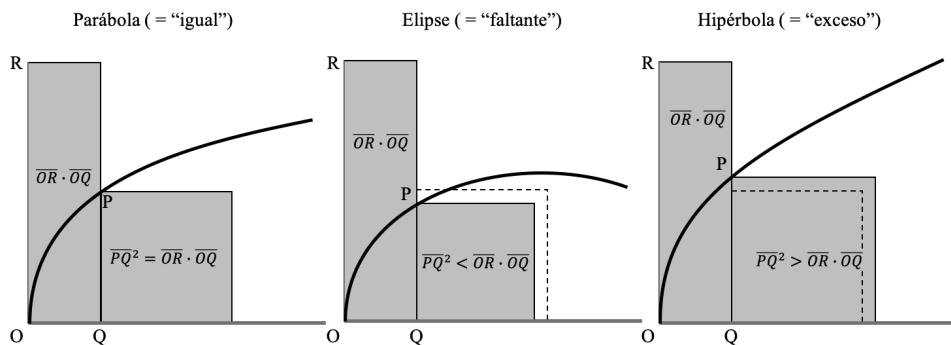


Figura 4.8: Definición con áreas aplicadas de las curvas cónicas y el origen de sus nombres.

con un plano perpendicular a su eje de simetría. Si el plano, sin embargo, forma un ángulo distinto de 90 grados con el eje de simetría, pero no es paralelo a los lados del cono, la figura resultante, que es también cerrada como la circunferencia, se llama una **elipse**. Si el plano es paralelo a los lados oblicuos del cono la figura resultante es abierta y la llamamos una **parábola**. Finalmente, si el plano no es paralelo a los lados del cono, pero por su ángulo nunca intersecta el eje de simetría, decimos que la figura que se forma es una **hipérbola**.

4.2.2. Del nombre al álgebra

La palabra “parábola” viene del griego *παραβολή* (“parabalei”) que significa “poner al lado”, “igualar”, “comparar” (de allí que una parábola en español sea también una historia que donde se narran hechos que pueden servir como modelos de comportamiento.) Este curioso nombre tiene su origen en la manera como este tipo de curva fue definida de forma más rigurosa a como lo hicimos en la sección anterior, por uno de los más grandes matemáticos de la antiguedad, Apolonio de Perga (ca. 262 a ca. 190 a.e.c.) en su tratado clásico *Conicas*.

En la [Figura 4.8](#) se ilustra la construcción de Apolonio y la razón para el nombre que dió a cada cónica. Para ello ubicamos las curvas sobre el plano de modo que su eje de simetría quede alineado con una semirrecta horizontal que comienza en el punto O que llamaremos *apside* de la cónica (la elipse tiene dos *apsides*.) Todas las cónicas de la figura tienen asociado un parámetro que define su tamaño y que es igual a la longitud de un segmento \overline{OR} perpendicular al eje de simetría. Si aumentamos o disminuimos la longitud de este segmento las cónicas serán más grandes o más pequeñas de las representadas en la figura. Por cada punto P de las cónicas existe un punto Q que es su proyección sobre el eje de simetría.

La propiedad descubierta por Apolonio (y posiblemente por Euclides y otros matemáticos anteriores a él) es que el área del rectángulo que tiene como base el segmento \overline{OQ} y como altura el parámetro de tamaño, es decir, la longitud del segmento \overline{OR} puede ser (dependiendo de la cónica) igual, menor o mayor al área del cuadrado que tiene como lado la longitud del segmento \overline{PQ} . En la geometría clásica a esta operación se la llama la *aplicación de un rectángulo* o la *cuadratura del*

rectángulo.

La parábola es entonces la figura en la que esas áreas son exactamente iguales. De allí su nombre en griego. Por otra parte la elipse, cuyo nombre viene del griego *ελειπεῖν* ("eleipeín") que significa "faltante", es tal que el área del cuadrado no alcanza a ser igual a la del rectángulo. Y en la hipérbola, cuyo nombre viene del griego *νπερβαλλεῖν* ("iperbalein") que significa "exceso", el área del cuadrado supera la del rectángulo aplicado.

En términos algebraicos (geometría analítica), si construimos un sistema de coordenadas cartesianas con origen en el ápice O, eje x_a (el subíndice indica precisamente que el origen está en el ápice) en la dirección del eje de simetría y eje y_a en dirección del semieje \overline{OR} (cuya longitud llamaremos L), las coordenadas (x_a, y_a) de los puntos sobre las cónicas se relacionan de acuerdo con [4]:

$$y_a^2 = Lx_a + \eta x_a^2 \quad (4.64)$$

Aquí η es un parámetro que define la "forma" de la cónica, siendo:

- $\eta = 0$ en el caso de un **parábola**.
- $\eta < 0$ en el caso de una **elipse**.
- $\eta > 0$ en el caso de una **hipérbola**.

Por su origen llamaremos a η el parámetro de Apolonio de la cónica.

En el algoritmo a continuación usamos estas definiciones para dibujar las cónicas y ver el efecto que tienen los parámetros L y η en su tamaño y forma.

(Algoritmo 4.12)

```
#Definimos el algoritmo como una rutina
def conicas_apolonio(eta=-0.5):

    #Escala
    L=10.0
    #Forma
    eta=float(eta)

    #Máximo valor de x
    xamax=L/abs(eta) if abs(eta)>0 else L

    #Valores de x en los que graficaremos
    from numpy import sqrt,linspace
    xas=linspace(0,xamax,100)

    #Ecuaciones de las cónicas referidas al apside
    from numpy import append
    yas_par=sqrt(L*xas)
    yas=sqrt(L*xas+float(eta)*xas**2)

    #Gráfica
    import matplotlib.pyplot as plt
    fig=plt.figure(figsize=(6,6))
    ax=fig.gca()
```

```

ax.plot(xas,yas_par,'k--')
ax.plot(xas,-yas_par,'k--')
ax.plot(xas,yas,'b')
ax.plot(xas,-yas,'b')

#Decoración
ax.grid()
ax.set_title(f"Cónica con $L = {L}$, $\eta={eta}$")

#Fijamos la misma escala en los ejes
from pymcel.plot import fija_ejes_proporcionales
valores=(xas,yas_par,-yas_par,yas,-yas),
fija_ejes_proporcionales(ax,valores,
                         xmin=-xamax/2,ycm=0);

#Invocamos la rutina
conicas_apolonio()

```

ver Figura 4.9

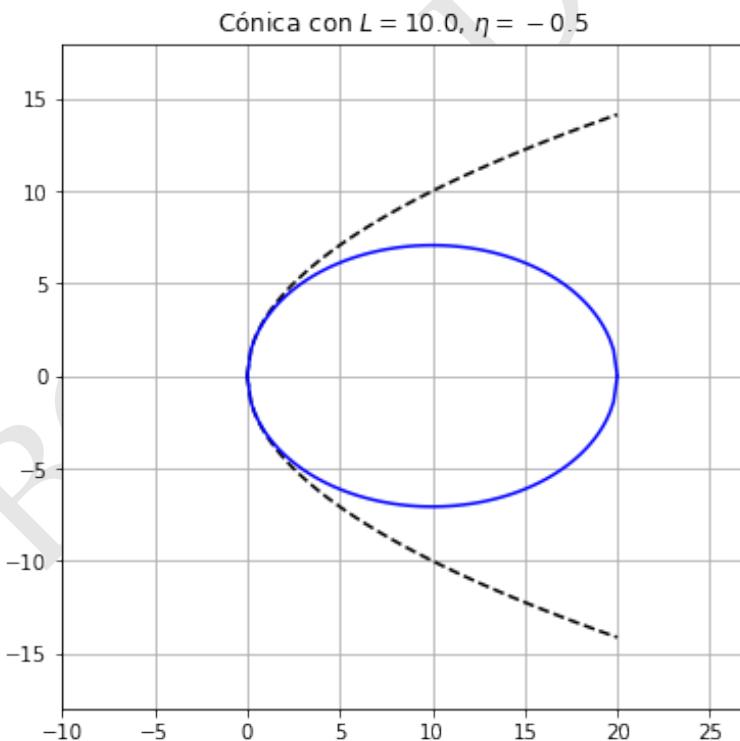


Figura 4.9: Figura correspondiente al código 4.12.

Trate de demostrar usando el algoritmo o manipulando la Ec. (4.64) que la circunferencia, que en la definición original se considera una cónica más, no es más

que una elipse para la cuál el parámetro de forma η es igual a -1.

Para ver esta una versión interactiva de esta figura por favor use las libretas disponibles en la [versión electrónica del libro](#)²².

4.2.3. Directriz de las cónicas

La definición de Apolonio de las cónicas provista en la sección anterior, no solo nos permite introducir la primera fórmula algebraica para describirlas (Ec. ??) y tal vez la primera en la historia que lo hizo, sino que además nos ayudó a entender el origen de sus nombres.

Sin embargo, esta definición adolece de algunas características que nos resultaran muy útiles en lo sucesivo. No es claro, por ejemplo, la relación entre los parámetros de tamaño L y forma η con otras propiedades de la cónica (en el caso de una elipse por ejemplo con su diámetro.)

Una definición más conveniente es la de Arquímedes (introducida posiblemente unos años antes que Apolonio y usada más frecuentemente en la edad media y tiempos modernos.) Esta definición se basa en proporciones más que en áreas.

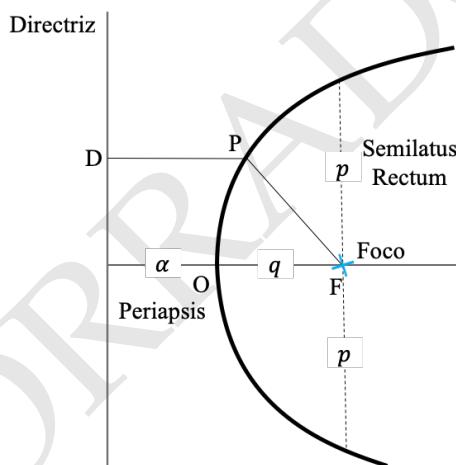


Figura 4.10: Definición de las cónicas usando la recta directriz y el foco.

Según esta definición, toda cónica se puede construir partiendo de una recta a la que se llama *recta directriz* y un punto o *foco* (ver Figura 4.10.) Las cónicas son el lugar geométrico de los puntos tal que la razón de la distancia del punto al foco y la distancia del punto a la directriz es constante:

$$\frac{\overline{PF}}{\overline{PD}} \equiv e \quad (4.65)$$

donde $e > 0$. A este parámetro lo llamamos la *excentricidad* de la cónica.

Si definimos un sistema de coordenadas tal que el eje y_d pase por la recta directriz (el subíndice d indica que el sistema de coordenadas esta precisamente referido

²²<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

a la directriz) y el eje x_d por el foco, en términos de las coordenadas (x_d, y_d) de cada punto de la cónica, la condición en la Ec. (4.65) se puede escribir como:

$$\frac{\sqrt{(F - x_d)^2 + y_d^2}}{x_d} = e \quad (4.66)$$

donde F es la distancia del Foco a la directriz.

Despejando y obtenemos:

$$y_d^2 = (e^2 - 1)x_d^2 + 2Fx_d - F^2 \quad (4.67)$$

Una comparación de esta ecuación con la Ec. (4.64) revela inmediatamente que el valor de la excentricidad dependerá de la cónica así:

- $e = 1$ en el caso de la **parábola**.
- $0 < e < 1$ en el caso de una **elipse**.
- $e > 1$ en el caso de una **hipérbola**.

Estudiemos ahora la posición de dos puntos sobre la cónica que merecen alguna atención. El primero es el punto que se encuentra justo encima del foco, es decir $x = F$. La distancia vertical del eje de simetría a este punto se conoce como el *semilatus rectum* p y se denota como p . Reemplazando en la Ec. (4.67):

$$p^2 = (e^2 - 1)F^2 + 2F^2 - F^2$$

de donde se obtiene:

$$p = eF$$

El otro punto es el apside más cercano a la directriz. Si llamamos α a la distancia de ese punto a la directriz y usamos la definición de la cónica en la Ec. (4.66) obtenemos:

$$\alpha = \frac{F}{1 + e} \quad (4.68)$$

Por otro lado la distancia del foco al apside más cercano, que llamaremos **periapsis** q , viene dada en términos de F, e por:

$$\begin{aligned} q &= F - \alpha \\ &= eF / (1 + e) \end{aligned}$$

o bien,

$$q = \frac{p}{1 + e} \quad (4.69)$$

Si trasladamos ahora la Ec. (4.67) el origen de coordenadas en el punto O, es decir si reemplazamos $x_d = x_a + \alpha$, $y_d = y_a$ obtenemos:

$$y_a^2 = 2eFx_a + (e^2 - 1)x_a^2 \quad (4.70)$$

que comparando con la Ec. (4.64) nos permite escribir los parámetros de tamaño L y forma η en la definición de Apolonio, en función de la excentricidad e y *semilatus rectum* p :

$$L = 2p$$

$$\eta = e^2 - 1 \quad (4.71)$$

De esta última relación y de la frase final de la Sección 4.2.2 en la que identificamos a la circunferencia como la cónica con $\eta = -1$, concluimos nuevamente que la circunferencia no es más que una elipse para la cuál $e = 0$.

4.2.4. Síntesis geométrica

En resumen podemos decir que las curvas cónicas, o al menos sus principios de construcción, eran conocidas en Grecia posiblemente desde el tiempo de los pitagóricos. Sin embargo, fue Apolonio de Perga quién profundizó en las propiedades geométricas y “algebraicas” de estas curvas. Su aplicación astronómica, sin embargo, era desconocida y fue descubierta solo hasta los trabajos de Kepler en los 1600.

En términos geométricos simples las cónicas con el resultado de la intersección de un cono con un plano con cuatro posibles inclinaciones respecto a su eje de simetría. A pesar de que esto implicaría que existen cuatro posibles cónicas, las definiciones posteriores demuestran que en realidad la circunferencia es un caso particular de la elipse ($\eta = -1$ o $e = 0$).

Toda cónica tiene: un eje de simetría, una recta directriz, un foco, un ápide (el periapsis o punto más cercano al foco) y un *latus rectum* (segmento perpendicular al eje y que pasa por el foco.) Las elipses tienen un segundo eje de simetría, perpendicular al primero y a distancias iguales de dos apses y por la misma razón un segundo foco y un segundo *latus rectum*.

Las dimensiones y forma de una cónica se pueden describir, en general, en términos de dos parámetros:

- **Parametrización de Apolonio.** Los parámetros son la longitud del *latus rectum* $L > 0$ y el parámetro de Apolonio $-\infty < \eta < \infty$. De acuerdo con η las cónicas se clasifican en: parábola ($\eta = 0$), circunferencia o elipse ($\eta < 0$) e hipérbola ($\eta > 0$).
- **Parametrización de Arquímedes.** Los parámetros son el *semilatus rectum* $p > 0$ y la excentricidad $0 \leq e < 1$. De acuerdo con e las cónicas se clasifican en: parábola ($e = 1$), circunferencia y elipse ($0 \leq e < 1$) e hipérbola ($e > 1$).

Se acostumbra usar más la parametrización de Arquímedes y es la que utilizaremos en lo sucesivo en el libro.

Si se construye un sistema de coordenadas en el plano de la cónica, con eje x en la dirección del eje de simetría, dependiendo de la localización del origen, las coordenadas cartesianas de los puntos sobre la cónica obedecen las ecuaciones:

- **Origen en el ápide:** $y_a^2 = 2px_a + (e^2 - 1)x_a^2$.
- **Origen en la directriz:** $y_d^2 = (e^2 - 1)x_d^2 + (2p/e)x_d - (p/e)^2$

Esta última ecuación no es conveniente para describir la circunferencia.

4.2.5. Descripción algebraica

Si bien en las secciones anteriores adscribimos a Apolonio y a Arquímedes (y a sus contemporáneos griegos) la descripción de las cónicas en términos de ecuaciones algebraicas con sus coordenadas como variables, en realidad esta descripción solo apareció en la historia con el surgimiento de la moderna Geometría analítica en los 1600 y de la mano de René Descartes (“René decart”²³) y Pierre de Fermat (“pier de fermat”).

Ahora bien, la forma algebraica general que vimos en la Ec. (4.64)

$$y_a^2 - 2px_a - (e^2 - 1)x_a^2 = 0 \quad (4.72)$$

no es precisamente ni la fórmula más simple, ni la más general.

En esta sección exploraremos a fondo las propiedades algebraicas de las cónicas y al hacerlo descubriremos algunas propiedades importantes que usaremos en la mecánica celeste.

Si queremos escribir la ecuación de la cónica en otras formas, podemos, como lo hicimos en el caso de la Eq. 4.70, aplicar dos tipos de transformaciones al sistema de coordenadas:

- Una **traslación**, que implica simplemente modificar la posición del origen de coordenadas, tal y como hicimos en la Sección 4.2.3 al pasar del origen en la directriz a un origen en el ápide.
- Una **rotación**, que implica modificar la dirección de los ejes coordenados. Esta es una transformación más compleja pero que resultará particularmente útil en la aplicación de las cónicas en mecánica celeste.

4.2.6. Ecuación respecto al centro

Para encontrar una forma más simple de la ecuación de la cónica, podemos realizar una traslación del origen, del apside (respecto al cuál esta escrita la Ec. 4.64) a un punto C en el cuál la forma algebraica, por ejemplo solo contenga términos cuadráticos en las coordenadas.

Una traslación a lo largo del eje x se escribe como:

$$\begin{aligned} x_a &= x_c + a \\ y_a &= y_c \end{aligned} \quad (4.73)$$

donde a es la distancia del origen de las nuevas coordenadas x_c, y_c (punto C), al origen de las coordenadas x_a, y_a (apside).

Reemplazando las Ecs. (4.73) en la ecuación de la cónica obtenemos:

$$y_a^2 - 2p(x_c + a) + (1 - e^2)(x_c + a)^2 = 0$$

que después de una manipulación algebraica se puede escribir en la forma:

$$y_c^2 + (1 - e^2)x_c^2 + 2[(1 - e^2)a - p]x_c = 2pa - (1 - e^2)a^2$$

²³https://es.forvo.com/word/ren%C3%A9_descartes/#fr

Como el valor de a es libre, podemos escogerlo de modo el término lineal en x_c desaparezca. Con esta elección el valor de esta constante queda:

$$a = \frac{p}{1 - e^2} \quad (4.74)$$

Descubrimos aquí que el desplazamiento al punto C solo tiene el efecto deseado en el caso en el que $e \neq 1$ (elipse e hipérbola). Para el caso de la parábola, en realidad la forma más simple de la ecuación sigue siendo aquella referida al apside (Ec. 4.64), $y_d^2 = 2px_d$.

La ecuación de la elipse o de la hipérbola se puede escribir entonces como:

$$\frac{x_c^2}{p^2/(1-e^2)^2} + \frac{y_c^2}{p^2/(1-e^2)} = 1$$

o en términos del parámetro a :

$$\frac{x_c^2}{a^2} + \frac{y_c^2}{b^2} = 1 \quad (4.75)$$

en el que se ha definido:

$$b^2 \equiv a^2(1 - e^2) \quad (4.76)$$

4.2.7. Eje mayor y menor de la elipse

La ecuación (4.75) es la forma más simple y simétrica de una elipse o una hipérbola. Pero ¿qué interpretación geométrica tienen los parámetros a , b y el punto C .

En el caso de la elipse, podemos ver que cuando $y_c = 0$ (apsides), $x_c = \pm a$ (ver Figura 4.12), es decir a es la distancia de los apseses al punto C , alrededor del cuál la elipse es simétrica. Llamamos a la constante a en este caso el **semieje mayor** de la elipse y C es el centro geométrico de la figura.

También en el caso de la elipse, haciendo $x_c = 0$ resulta $y_c = \pm b$, de donde interpretamos a $b = a\sqrt{1 - e^2}$ como la distancia a los extremos del *eje menor* al centro de la elipse. Por esta misma razón llamamos a la constante b el **semieje menor**.

Un resultado interesante, que podemos agregar a las definiciones geométricas de Apolonio y Democrito, resulta al combinar la definición de a con la distancia de un apside al foco más cercano q . Es claro de la ?? que la distancia entre el centro y el foco de la elipse, que llamaremos en lo sucesivo c es:

$$\begin{aligned} c &= a - q \\ &= \frac{p}{1 - e^2} - \frac{p}{1 + e} \end{aligned}$$

o en términos simples:

$$c = ae \quad (4.77)$$

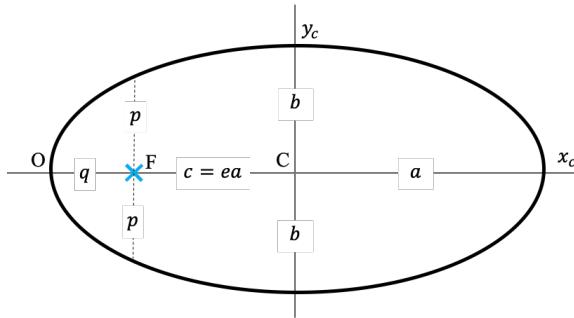


Figura 4.11: Parámetros geométricos de la elipse referidos al apside O, el foco F y el centro C: a semieje mayor, b semieje menor, p semilatus rectum, e excentricidad, c distancia foco-centro.

Este resultado se puede interpretar diciendo que la excentricidad de una elipse $e = c/a$ es el grado en el que el foco esta desplazado a partir del centro geométrico de la figura y medido en unidades del semieje mayor a . De allí precisamente el nombre de este parámetro.

Uno podría entonces cuantificar la excentricidad de una elipse como un porcentaje. Así por ejemplo, la órbita osculatriz de Marte (volveremos sobre este concepto en el ??), que fue el planeta utilizado por Kepler para descubrir que las órbitas planetarias eran elipses, es una elipse con una excentricidad de 9,3 %.

Es decir, el centro geométrico de la órbita de Marte esta desplazado respecto a su foco (donde se ubica el Sol o más precisamente el centro de masa del Sistema Solar) un 9,3 % del eje mayor (la distancia promedio de Marte al Sol.) Este desplazamiento (que es significativo) fue la clave precisamente de porque fue más fácil para Kepler deducir la elipticidad de la trayectoria de ese planeta, de lo que lo fue para todos los astrónomos antes que él deducirlo usando la trayectoria de todos los demás planetas (en comparación las excentricidades de las órbitas osculatrices de la Tierra y de Júpiter, por ejemplo, son 1,6 % y 4,8 %, respectivamente.)

Utilice el código interactivo que viene con las libretas en la [versión electrónica del libro²⁴](#) para visualizar la forma de las órbitas de los planetas mencionados y ver realmente, que tan diferentes de una circunferencia son.

4.2.8. Parámetros de la hipérbola

La interpretación geométrica de a y b en el caso de la hipérbola ($e > 1$) es un poco más complicada (ver la ??).

Para empezar el valor del parámetro $a = p/(1 - e^2)$ es negativo. Esto implica que el punto C, que hace la ecuación de la hipérbola la más sencilla posible, esta a la izquierda del apside, contrario a lo que pasa con la elipse. Llamamos a C, no el centro de la hipérbola sino su vértice.

No debemos confundir sin embargo el eje y_c en la ?? con la directriz de la hipérbola (ver Figura 4.10) que en realidad esta situada a una distancia $\alpha = a(1 - e)/e$ (ver Ec. 4.68.) La directriz y el eje y_c se aproximan una a otra cuando $e \rightarrow \infty$.

²⁴<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

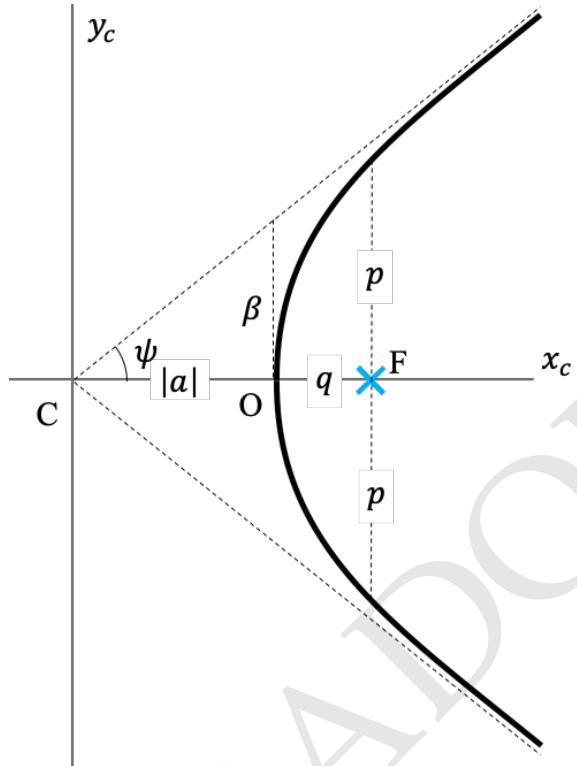


Figura 4.12: Parámetros geométricos de la hipérbola referidos al apside O , el foco F y el vértice C : a distancia al vértice (llamado con frecuencia también semieje mayor aunque en la hipérbola no hay tal), β pendiente de la hipérbola, p semilatus rectum, e excentricidad, ψ angulo de semiapertura.

El parámetro $b^2 = a^2(1 - e^2)$ es inconveniente dado que en este caso $e > 1$ y por lo tanto b debería ser un número imaginario. Para evitar este inconveniente reescribimos la ecuación de la hipérbola Ec. (4.75) como:

$$\frac{x_c^2}{a^2} - \frac{y_c^2}{\beta^2} = 1 \quad (4.78)$$

y definimos:

$$\beta \equiv |a| \sqrt{e^2 - 1} \quad (4.79)$$

¿Cuál es la interpretación geométrica de β ? Si despejamos y_c de la Ec. (4.78) tenemos:

$$y_c = \pm \beta \sqrt{\frac{x_c^2}{a^2} - 1}$$

donde los signos \pm corresponden a las “ramas” superior e inferior de la hipérbola.

Aquí descubrimos una interesante propiedad de esta cónica: cuando $x_c \rightarrow \infty$, la hipérbola se aproxima a las rectas:

$$y_c \rightarrow \pm \frac{\beta}{|a|} x_c$$

que llamamos *asíntotas*. La pendiente de las asíntotas, $\beta/|a|$ nos permite identificar a β como una cantidad que cuantifica el grado de *apertura* de la hipérbola respecto a su eje de simetría: a mayor β (mayor excentricidad), mayor es la pendiente de las asíntotas y más cerca está la hipérbola de una línea recta paralela a la directriz.

Otra manera de cuantificar la pendiente de las asíntotas es usar el ángulo ψ :

$$\tan \psi \equiv \frac{\beta}{|a|}$$

No es difícil mostrar (ver problemas al final del capítulo), a partir de la definición anterior, que:

$$\cos \psi = \frac{1}{e} \quad (4.80)$$

Para poner en un contexto astronómico este resultado, podemos mencionar que en 2019 fue descubierto un cometa proveniente del espacio interestelar, hoy conocido como el 2I/Borisov, cuya trayectoria respecto al centro de masa del sistema solar es una hipérbola con $e = 3,35$. Usando la Ec. (4.80) podemos calcular que las asíntotas de su órbita se abren en un ángulo extremo de

```
#Cometa interestelar 2I/Borisov
e=3.35

from numpy import arccos
psi=arccos(1/e)

Psi = 72.63202008811639 grados
```

Utilice el código interactivo que viene con las libretas en la [versión electrónica](#) del libro²⁵ para visualizar la forma de la órbita del 2I/Borisov.

4.2.9. Rotación de las cónicas en el plano

Al comenzar la ??, habíamos mencionado dos posibles transformaciones que nos conducían a la forma algebraica más simple (la que obtuvimos en la [Sección 4.2.6](#)) y a la más general. La primera la obtuvimos simplemente aplicando una traslación de los ejes coordenados:

$$\begin{aligned} x_c &= x_a - c \\ y_c &= y_a \end{aligned}$$

Para obtener la forma más general, nos proponemos ahora realizar una rotación.

Si llamamos x' , y' , z' a las coordenadas de un punto con respecto a los ejes rotados, puede demostrarse (ver problemas al final del capítulo) que estas se relacionan con las coordenadas del mismo punto en el sistema alineado con la cónica (eje x en dirección del eje de simetría) a través de:

²⁵<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

$$\begin{aligned}x' &= x \cos \theta + y \sin \theta \\y' &= -x \sin \theta + y \cos \theta \\z' &= z\end{aligned}\quad (4.81)$$

donde θ es el ángulo que forma el eje x' con el eje x (ver ??.)

Matricialmente estas relaciones se pueden escribir como:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = R_z(\theta) \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (4.82)$$

donde la matriz $R_z(\theta)$ esta dada por:

$$R_z(\theta) = \begin{pmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (4.83)$$

y el subíndice z indica que la rotación se realiza alrededor de este eje.

La matriz de rotación $R_z(\theta)$ es una matriz *unitaria* que tiene las siguientes propiedades:

- Determinante, $\det R_z = 1$.
- Inversa, $R_z^{-1} = R_z^T$

Esta última propiedad implica que:

$$R_z^{-1}(\theta) = R_z(-\theta) \quad (4.84)$$

que será muy conveniente para lo que viene.

Usando las propiedades de R_z podemos encontrar la transformación inversa a la Ec. (4.81):

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = R_z(-\theta) \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \quad (4.85)$$

o explícitamente:

$$\begin{aligned}x &= x' \cos \theta - y' \sin \theta \\x &= x' \sin \theta + y' \cos \theta \\z &= z'\end{aligned}\quad (4.86)$$

El sistema SPICE contiene una rutina útil para definir de manera sencilla una matriz de rotación dado un ángulo y un eje respecto al que se realiza la rotación:

(Algoritmo 4.13)

```
from spiceypy import rotate
from numpy import pi
Rz=rotate(pi/6,3)

Rz(30 grados) =
[[ 0.866  0.5    0.    ]
 [-0.5     0.866  0.    ]
 [ 0.      0.      1.    ]]
```

Que coincide con la definición dada por la Ec. (4.83).

4.2.10. Ecuación general de las cónicas

Qué pasa entonces si, partiendo de la ecuación general respecto al apside (Ec. 4.72) hacemos primero una traslación a un punto con coordenadas (t_x, t_y) :

$$(y + t_y)^2 - 2p(x + t_x) - (e^2 - 1)(x + t_x)^2 = 0$$

y una vez trasladados al nuevo origen, realizamos una rotación a unos nuevos ejes (x', y') realizando para ello la transformación dada por las Ecs. (4.86):

$$[(x' \sin \theta + y' \cos \theta) + t_y]^2 - 2p[(x' \cos \theta - y' \sin \theta) + t_x] - (e^2 - 1)[(x' \cos \theta - y' \sin \theta) + t_x]^2 = 0$$

Expandiendo y recogiendo términos comunes, la ecuación general de una cónica trasladada y rotada será:

$$\begin{aligned} x'^2(1 - e^2 \cos^2 \theta) + x'y'(e^2 \sin 2\theta) + y'^2(1 - e^2 \sin^2 \theta) &+ \\ x'[2t_y \sin \theta - 2p \cos \theta + 2t_x \cos \theta(1 - e^2)] &+ \\ y'[2t_y \cos \theta + 2p \sin \theta - 2t_x \sin \theta(1 - e^2)] &+ \\ t_x^2(1 - e^2) - 2pt_x + t_y^2 &= 0 \end{aligned} \quad (4.87)$$

que puede escribirse de forma general, como:

$$Ax'^2 + Bx'y' + Cy'^2 + Dx' + Ey' + F = 0 \quad (4.88)$$

con:

$$\begin{aligned} A &= 1 - e^2 \cos^2 \theta \\ B &= e^2 \sin 2\theta \\ C &= 1 - e^2 \sin^2 \theta \\ D &= 2t_y \sin \theta - 2p \cos \theta + 2t_x \cos \theta(1 - e^2) \\ E &= 2t_y \cos \theta + 2p \sin \theta - 2t_x \sin \theta(1 - e^2) \\ F &= t_x^2(1 - e^2) - 2pt_x + t_y^2 \end{aligned} \quad (4.89)$$

Es decir, cualquier curva en el plano cuyos puntos obedezcan una ecuación cuadrática general de la forma Ec. (4.88) es una cónica con una orientación, tamaño, forma y posición de los apsides que dependerá de los coeficientes A, B, C, D, E y F .

Dada la ecuación algebraica de una cónica, expresada en la forma cuadrática general (Ec. 4.88), es posible, usando los coeficientes A, B y C determinar qué tipo de cónica y su orientación en el espacio.

Para ello es posible, combinando algunas de las Ecs. (??), demostrar que:

$$\eta = 1 - (A + C) \quad (4.90)$$

y por otro lado:

$$\tan 2\theta = \frac{B}{C - A} \quad (4.91)$$

Si además se usan los valores de los coeficientes D y E , es posible determinar la ordenada del vértice de la cónica:

$$t_y = \frac{D \sin \theta + E \cos \theta}{2} \quad (4.92)$$

Expresiones mucho más complejas pueden derivarse para t_x y para p en función de los coeficientes D , E y F , que también dependen de ellos (ver problemas al final del capítulo.) Sin embargo, una vez el apside de la cónica ha sido localizada sobre el eje de las abcisas (realizando la traslación inversa $-t_y$) y ha sido rotada en un ángulo θ dado por la Ec. (4.91) para que su eje de simetría coincida con el eje x , las demás propiedades de la curva pueden obtenerse más fácilmente.

4.2.11. Gráfico de una cónica rotada en el plano

Podemos poner en práctica algunos de los resultados de las secciones anteriores (siempre es importante hacerlo para garantizar que todo se ha entendido bien), construyendo numéricamente una cónica con la Ec. (4.72) y aplicando traslaciones y rotaciones a la misma para ver el efecto y escribir la ecuación general cuadrática (Ec. 4.88) que la describe.

Comencemos, escogiendo las propiedades de nuestra cónica y determinando, el rango de valores de las coordenadas x_a de los puntos sobre la cónica, en el sistema de coordenadas que tiene origen en el apside:

```

p=10.0
e=0.8

#Parametro eta
eta=e**2-1

#Conjunto de valores de x de la cónica
from numpy import linspace
if e==1:
    a=p
else:
    a=p/(1-e**2)

#Rango de valores de x
xs=linspace(0,2*abs(a),100)

```

Nótese que a no está definido en el caso de una parábola $e = 1$ (para una elipse y una hipérbola $a = p/(1 - e^2)$, ver Ec. 4.74.) Por esa razón en el algoritmo anterior hemos escogido definir $a \equiv p$ cuando $e = 1$, únicamente con el propósito de usar una sola expresión $xs=linspace(0,2*abs(a),100)$ para calcular el rango de valores de las abcisas de la cónica.

Si bien sabemos que en el caso de paráolas e hipérbolas, los valores de las abcisas son $x_a \in [0, \infty)$ (ver Figura 4.8), para una elipse es claro que $x_a \in [0, 2a]$ (ver Figura 4.12.)

Con los valores de x podemos ahora usar la Ec. (4.72) para calcular los valores de la ordenada, tanto para la rama superior de la cónica (por encima del eje de

simetría) como para la inferior:

```
#Ecuación de la cónica
from numpy import sqrt
ys_sup=sqrt(2*p*xs+eta*xs**2)
ys_inf=-sqrt(2*p*xs+eta*xs**2)
```

Para los propósitos de graficar la cónica, debemos duplicar los valores de la lista `xs`. Sin embargo, y por razones que veremos abajo, lo haremos ordenando de forma especial las coordenadas de los puntos, comenzando primero con el punto más lejano al origen ($x = 2a$ o $xs[-1], ys[-1]$), llegando hasta el origen mismo $xs[0], ys[0]$ y de allí regresando de nuevo al punto más lejano.

En Python esta “compleja” operación puede abreviarse usando la sintaxis general para extraer tajadas de los arreglos:

```
from numpy import append,zeros_like
xs=append(xs[::-1],xs)
ys=append(ys_sup[::-1],ys_inf)
zs=zeros_like(xs)
```

Un gráfico de los puntos de la cónica en el sistema de referencia del ápside srá:

(Algoritmo 4.14)

```
import matplotlib.pyplot as plt
fig=plt.figure()
ax=fig.gca()

ax.plot(xs,ys, 'b-')

from pymcel.plot import fija_ejes_proporcionales
valores=(xs,ys)
fija_ejes_proporcionales(ax,valores,xmin=0,ycm=0);
ax.grid()
```

ver Figura 4.13

Ahora podemos trasladar la cónica:

```
#Parámetros de la traslación
tx=15.0
ty=-10.0

#Coordenadas trasladadas
xs=xs-tx
ys=ys-ty
```

Construimos la matriz de rotación usando las rutinas de SPICE (ver Alg. 4.13):

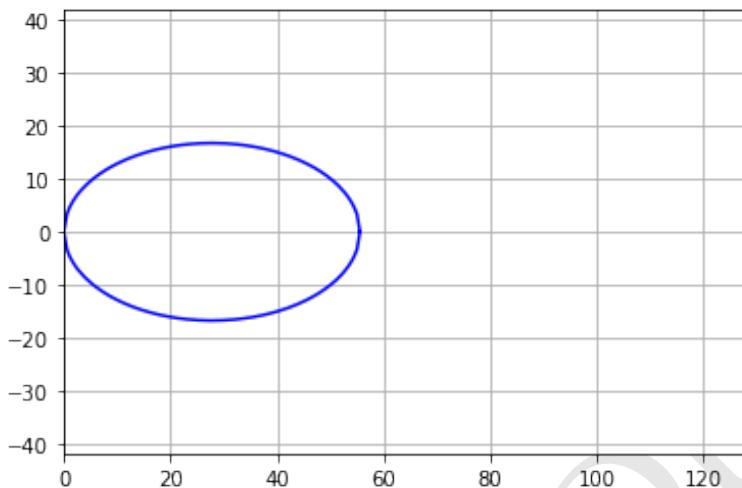


Figura 4.13: Figura correspondiente al código 4.14.

```
from spiceypy import rotate
from numpy import pi
teta=pi/6
Rz=rotate(teta,3)
```

La rotación de los puntos de la cónica contenidos en las listas `xs`, `ys` usando la Ec. (4.82), no es tan trivial en Python. Para ello definiremos una rutina general que usaremos más adelante en el libro:

(Algoritmo 4.15)

```
def rota_puntos(R,x,y,z):
    from spiceypy import mxv
    from numpy import zeros_like
    N=len(x)
    xp=zeros_like(x)
    yp=zeros_like(y)
    zp=zeros_like(z)
    for i in range(N):
        xp[i],yp[i],zp[i]=mxv(R,[x[i],y[i],z[i]])
    return xp,yp,zp
```

Los puntos rotados de la cónica serán:

```
xps,yps,zps=rota_puntos(Rz,xs,ys,zs)
```

Y una gráfica de los puntos rotados:

(Algoritmo 4.16)

```

import matplotlib.pyplot as plt
fig=plt.figure()
ax=fig.gca()

ax.plot(xps,yps,'b-')

from pymcel.plot import fija_ejes_proporcionales
valores=(xps,yps)
fija_ejes_proporcionales(ax,valores);
ax.grid()

```

ver Figura 4.14

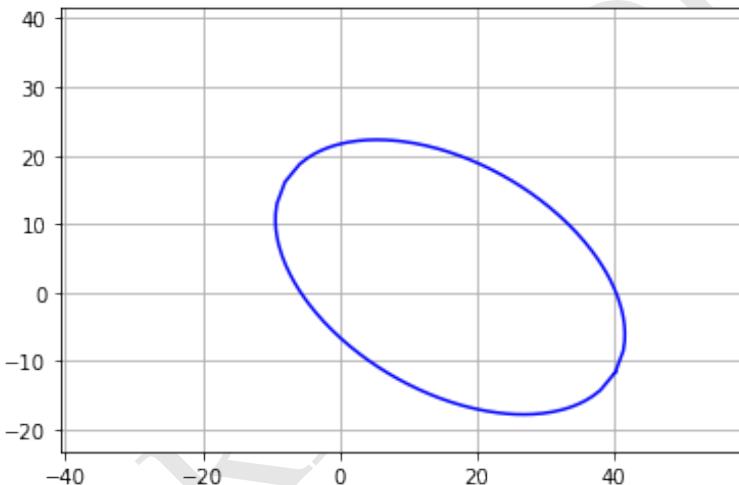


Figura 4.14: Figura correspondiente al código 4.16.

Los valores de los coeficientes del polinomio de segundo grado $P(x,y) = Ax^2 + By^2 + Cxy + Dx + Ey + F$ que describe los puntos del gráfico anterior, son (Ec. 4.89):

```

from numpy import sin,cos
A=1-e**2*cos(teta)**2
B=e**2*sin(2*teta)
C=1-e**2*sin(teta)**2
D=2*ty*sin(teta)-2*p*cos(teta)+2*tx*cos(teta)*(1-e**2)
E=2*ty*cos(teta)+2*p*sin(teta)-2*tx*sin(teta)*(1-e**2)
F=tx**2*(1-e**2)-2*p*tx+ty**2

```

A = 0.519999999999998

B = 0.5542562584220408

C = 0.84

D = -17.967433714816835

$E = -12.720508075688773$

$F = -119.00000000000003$

Ecuación: $(0.52)x^2 + (0.55)xy + (0.84)y^2 + (-18)x + (-13)y + (-119.0) = 0$

Podemos ahora verificar la afirmación que los puntos de la cónica satisfacen la ecuación $P(x, y) = 0$ (Ec. 4.88), construyendo primero una rutina para calcular, dado cualquier punto (x, y) el valor del polinomio $P(x, y)$:

(Algoritmo 4.17)

```
def polinomio_segundo_grado(coeficientes,x,y):
    A,B,C,D,E,F=coeficientes
    P=A*x**2+B*x*y+C*y**2+D*x+E*y+F
    return P
```

Si calculamos el valor de $P(x, y)$ para todos los puntos xps y yps verificamos la afirmación:

```
coeficientes=A,B,C,D,E,F
Pxpsyps=polinomio_segundo_grado(coeficientes,xps,yps)

print(f"P(xps,yps):\n{Pxpsyps[:5]}...")
```

```
P(xps,yps):
[ 0. -0. -0. -0. -0.]...
```

O para verificarlo en todos los puntos, podemos sumar el valor absoluto de los valores de $Pxpsyps$:

```
Psum=sum(abs(Pxpsyps))
```

Sum $|P(xps, yps)| = 1.1738165994756855e-11$

Finalmente podemos poner a prueba las Ecs. (??), (4.91) y (4.92):

```
#Parámetro de forma
eta_num=1-(A+C)

#Ángulo
from numpy import arctan,sin,cos
teta_num=0.5*arctan(B/(C-A))

#Desplazamiento vertical
ty_num=(D*sin(teta_num)+E*cos(teta_num))/2
```

```
eta original = -0.36, eta numérico = -0.36
teta original = 30, teta numérico = 30
ty original = -10, ty numérico = -10
```

Como era de esperarse, la coincidencia es perfecta.

4.2.12. Síntesis algebraica

En la [Sección 4.2.2](#) mostramos como las definiciones abstractas de la antigüedad, basadas en áreas y proporciones, se convirtieron en la edad media y el renacimiento, en ecuaciones algebraicas. Las representaciones algebraicas permiten describir de forma sintética y poderosa el lugar geométrico de las cónicas.

En las secciones precedentes hemos visto como la descripción algebraica de las cónicas depende de donde coloquemos el origen o en que dirección escojamos los ejes del sistema de coordenadas. De acuerdo a estas elecciones reconocimos hasta ahora 4 formas distintas de describir algebraicamente cualquier cónica, una vez especificados un parámetro de tamaño, por ejemplo el *semilatus rectum* p y uno de forma, por ejemplo la excentricidad e :

- Ecuación respecto al apside (origen en el apside, eje x sobre el eje de simetría, Ec. [4.64](#)):

$$y_a^2 - 2px_a - (e^2 - 1)x_a^2 = 0$$

- Ecuación respecto a la directriz (origen en la directriz, eje x sobre el eje de simetría, Ec. [??](#)):

$$y_d^2 - 2pex_d - (e^2 - 1)x_d^2 + e^2 p^2 = 0$$

- Ecuación respecto al centro (origen en el centro de simetría, eje x sobre el eje de simetría, Ec. [4.75](#), solo valida para $e \neq 1$):

$$\frac{x_c^2}{a^2} \pm \frac{y_c^2}{b^2} = 1$$

donde $a^2 = p/(1 - e^2)$, el signo “+” es para elipses ($e < 1$) en cuyo caso $b^2 = a^2(1 - e^2)$ y el signo “-” es para hipérbola ($e > 1$) para el cuál $b^2 = \beta^2 = a^2(e^2 - 1)$.

- Ecuación general (origen en t_x, t_y y eje x' rotado un ángulo θ respecto al eje de simetría, Ec. [4.88](#)):

$$Ax'^2 + Bx'y' + Cy'^2 + Dx' + Ey' + F = 0$$

con:

$$\begin{aligned} A &= 1 - e^2 \cos^2 \theta \\ B &= e^2 \sin 2\theta \\ C &= 1 - e^2 \sin^2 \theta \\ D &= 2t_y \sin \theta - 2p \cos \theta + 2t_x \cos \theta (1 - e^2) \\ E &= 2t_y \cos \theta + 2p \sin \theta - 2t_x \sin \theta (1 - e^2) \\ F &= t_x^2 (1 - e^2) - 2pt_x + t_y^2. \end{aligned}$$

4.2.13. Cónicas en coordenadas cilíndricas

Nos proponemos ahora a escribir la ecuación de la cónica, en coordenadas cilíndricas con origen en uno de los focos. La ecuación resultante y los resultados geométricos que se derivan de ella, es de primera importancia para la mecánica celeste.

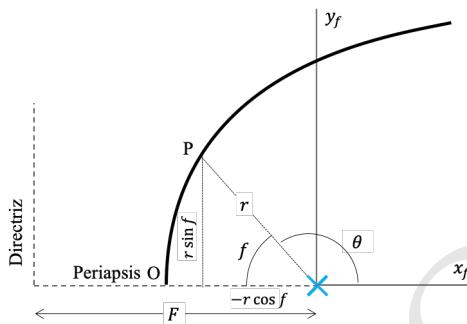


Figura 4.15: Derivación de la ecuación de la cónica en coordenadas cilíndricas referidas al Foco. En la figura el ángulo f es la anomalía verdadera.

Comenzando con la ecuación de la cónica referida a la directriz (Ec. 4.67) podemos aplicar una traslación al foco haciendo $x_d = x_f + F$ y $y_d = y_f$:

$$y_f^2 - (e^2 - 1)(x_f + F)^2 - 2F(x_f + F) + F^2 = 0 \quad (4.93)$$

Si escribimos x_f, y_f en coordenadas cilíndricas como (ver Ecs. 4.12):

$$\begin{aligned} x_f &= -r \cos f \\ y_f &= r \sin f \end{aligned} \quad (4.94)$$

donde f , en lugar de la coordenada cilíndrica acimutal convencional θ (que es un ángulo referido al semi eje $x+$) el ángulo entre la dirección del periapsis y el radio vector del punto (ver ??), la Ec. (4.93), después de algunas manipulaciones algebraicas se convierte en:

$$r = \frac{p}{1 + e \cos f} \quad (4.95)$$

que es la ecuación fundamental de la cónica y que veremos aparecer con mucha frecuencia en la mecánica celeste.

Las ecuaciones (??) y (4.95) evidencian un hecho interesante: el ángulo f se puede usar para describir matemáticamente, usando un sólo parámetro, las coordenadas cartesianas de los puntos sobre la cónica. Esto hace mucho más sencillo encontrar la posición de los puntos sobre estas curvas, en comparación como lo teníamos que hacer al usar las ecuaciones algebraicas en x, y de las secciones anteriores.

Para ilustrar el poder de este resultado considere el siguiente algoritmo para graficar una elipse y compárela con el visto en la ??:

(Algoritmo 4.18)

```

#Parámetros
p=10.0
e=0.8

#Valores del ángulo
from numpy import linspace,pi
fs=linspace(0,2*pi,100)

#Distancias
from numpy import cos
rs=p/(1+e*cos(fs))

#Coordenadas
from numpy import sin
xs=rs*cos(fs)
ys=rs*sin(fs)

#Gráfica
import matplotlib.pyplot as plt
fig=plt.figure()
ax=fig.gca()

#Puntos cónica
ax.plot(xs,ys,'b-')
#Foco
ax.plot([0],[0], 'cx', markersize=10)

#Decoración
ax.set_xlabel(f"${x}_f$")
ax.set_ylabel(f"${y}_f$")
from pymcel.plot import fija_ejes_proporcionales
valores=(xs,ys),
fija_ejes_proporcionales(ax,valores);
ax.grid();

```

ver Figura 4.16

Este será el método que utilizaremos en lo sucesivo para generar los puntos sobre cualquier cónica.

Un hecho interesante sobre el Alg. (4.18) es que no se generaliza fácilmente para el caso de una parábola o una hipérbola. La razón es que en los casos de conicas abiertas el valor del ángulo f esta limitado a un intervalo diferente al de la elipse en la que $f \in [0, 2\pi]$ o $f \in [-\pi, \pi]$.

En este caso si despejamos $\cos f$ de la Ec. (??):

$$\cos f = \frac{1}{e} \left(\frac{p}{r} - 1 \right)$$

cuando $r \rightarrow \infty$ el ángulo f adopta valores extremos dados por:

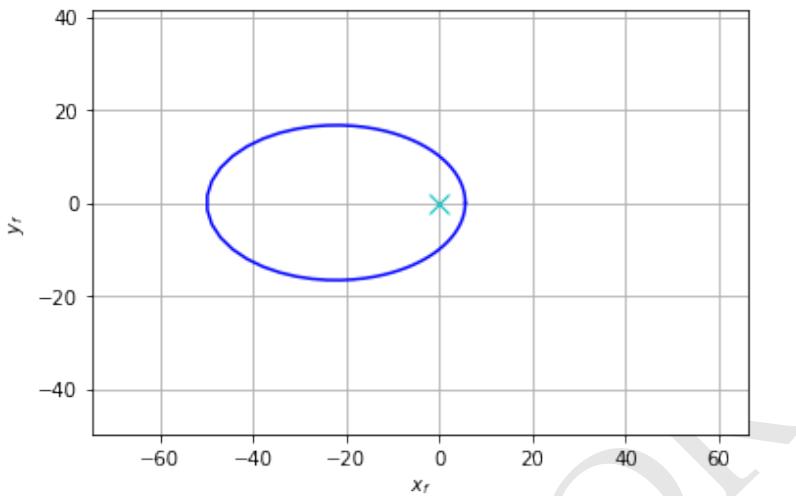


Figura 4.16: Figura correspondiente al código 4.18.

$$\cos f \rightarrow -\frac{1}{e}$$

Para el caso de la parábola esto implica que $f \in (-\pi, \pi)$, que es idéntico al caso de la elipse pero con el extremo inferior del intervalo abierto. En el caso de la hipérbola:

$$f \in (-\pi + \psi, \pi - \psi)$$

donde $\psi = \cos^{-1}(1/e)$ es el ángulo de apertura introducido en la Ec. (4.80).

Un algoritmo más general entonces, para generar los puntos sobre una cónica se presenta en la rutina a continuación:

(Algoritmo 4.19)

```
def puntos_conica(p,e,df=0.1):

    #Compute fmin,fmax
    from numpy import pi
    if e<1:
        fmin=-pi
        fmax=pi
    elif e>1:
        from numpy import arccos
        psi=arccos(1/e)
        fmin=-pi+psi+df
        fmax=pi-psi-df
    else:
        fmin=-pi+df
        fmax=pi-df
```

```

#Valores del ángulo
from numpy import linspace,pi
fs=linspace(fmin,fmax,500)

#Distancias
from numpy import cos
rs=p/(1+e*cos(fs))

#Coordenadas
from numpy import sin
xs=rs*cos(fs)
ys=rs*sin(fs)
from numpy import zeros_like
zs=zeros_like(xs)

return xs,ys,zs

```

Y un gráfico de la cónica, usando la rutina anterior sería:

(Algoritmo 4.20)

```

#Genera puntos
p=10.0
e=1.5
xs,ys,zs=puntos_conica(p,e)

#Gráfica
import matplotlib.pyplot as plt
fig=plt.figure()
ax=fig.gca()

ax.plot(xs,ys,'b-')
ax.plot([0],[0],'cx',markersize=10)

#Decoración
ax.set_xlabel(f"${x}_f$")
ax.set_ylabel(f"${y}_f$")
from pymcel.plot import fija_ejes_proporcionales
valores=(xs,ys),
fija_ejes_proporcionales(ax,valores);
ax.grid()

```

ver Figura 4.17

4.2.14. Anomalías

Además de las Ecs. (4.95) y (4.94) en las que describimos las coordenadas de los puntos sobre una cónica arbitraria como función de un único parámetro f (ecuaciones paramétricas), existe una segunda manera de expresar las ecuaciones de la elipse y de la hipérbola, en términos de otro parámetro.

En el caso de la elipse, por ejemplo, si partimos de la ecuación respecto al centro

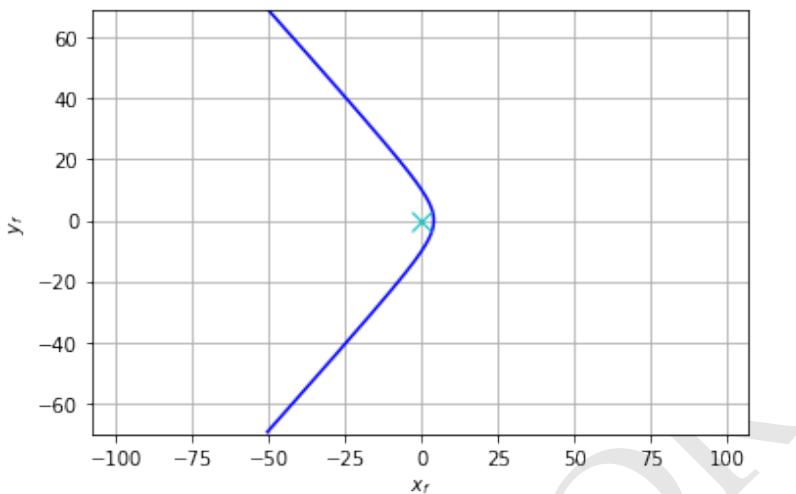


Figura 4.17: Figura correspondiente al código 4.20.

(Ec. 4.75):

$$\frac{x_c^2}{a^2} + \frac{y_c^2}{b^2} = 1$$

es posible escribir una forma parámetrica para las coordenadas:

$$\begin{aligned} x_c &= a \cos E \\ y_c &= b \sin E \end{aligned} \tag{4.96}$$

donde E es el nuevo parámetro.

La interpretación del parámetro f en la ecuación en coordenadas cilíndricas de la cónica era clara: el valor f para un punto dado, es al ángulo formado por la línea que va del foco al periapsis y la dirección del radio vector del punto. Por ser un ángulo que especifica la posición del punto respecto al foco (en el que se encuentra el Sol, en la teoría de Kepler del movimiento planetario), llamamos a f la **anomalía verdadera** del punto.

Un poco de historia

Kepler y las anomalías. El nombre de anomalías viene de Kepler.

¿Qué interpretación tiene por su parte el parámetro E en las Ecs. (4.96)?

En la construcción de la [Figura 4.18](#) identificamos a E como un nuevo ángulo, esta vez medido respecto al centro de la elipse y cuyo radio asociado al cortar dos círculos imaginarios de radios a y b , permiten encontrar la abcisa y la ordenada de los puntos de la elipse, respectivamente.

Por el hecho de medirse respecto al centro del círculo y no respecto del foco (en el que en la teoría de Kepler se encuentra el Sol, centro del Sistema Solar), llamamos a E la **anomalía excéntrica**.

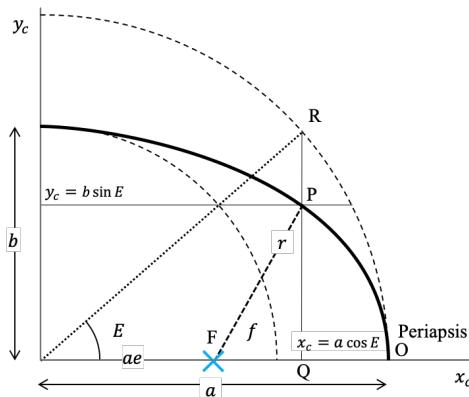


Figura 4.18: Definición de la anomalía excéntrica E y el método asociada a ella para determinar la posición de los puntos sobre una elipse.

Podemos escribir una ecuación para r en términos del parámetro E análoga a la Ec. (4.95) que nos da r en función de f ? ¡Sin duda alguna!

Consideré el triángulo entre los puntos FPQ en la Figura 4.18. El teorema de pitágoras en ese triángulo se escribe:

$$r^2 = (a \cos E - ae)^2 + b^2 \sin^2 E$$

Teniendo en cuenta que $b^2 = a^2(1 - e^2)$ y después de un poco de álgebra obtenemos:

$$r = a(1 - e \cos E) \quad (4.97)$$

que será una forma para representar la cónica alternativa a la Ec. (4.97) y que usaremos con frecuencia en el libro.

Finalmente, la manipulación adecuada de las ecuaciones anteriores permite escribir una relación explícita entre las anomalías verdadera f y excéntrica E que será muy utilizada a lo largo de este libro (ver problemas al final del capítulo):

$$\tan \frac{f}{2} = \sqrt{\frac{1+e}{1-e}} \tan \frac{E}{2} \quad (4.98)$$

Un procedimiento similar al anterior, pero en el caso de la hipérbola, permite escribir las coordenadas de los puntos de la curva en términos de un nuevo parámetro F :

$$\begin{aligned} x_c &= a \cosh F \\ y_c &= a \sinh F \end{aligned} \quad (4.99)$$

Por analogía con la elipse, F también es llamada la anomalía excéntrica (aunque en este caso la interpretación geométrica de F no es tan directa como en el caso de E .)

La distancia al foco se puede escribir en términos de F como:

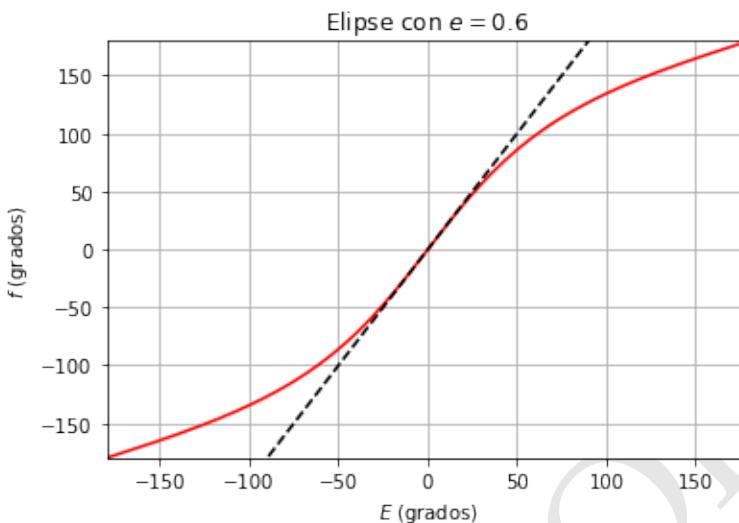


Figura 4.19: Anomalía verdadera f como función de la anomalía excéntrica E para una elipse. La línea rayada corresponde a la aproximación $f \approx \sqrt{(1+e)/(1-e)}E$ que es válida en el caso $f \ll 1$.

$$r = a(e \cosh F - 1) \quad (4.100)$$

y la relación entre la anomalía verdadera f y la anomalía excéntrica F resulta ser:

$$\tanh \frac{f}{2} = \sqrt{\frac{e+1}{e-1}} \tanh \frac{F}{2} \quad (4.101)$$

4.2.15. Área de las cónicas

Existe un último resultado de la geometría analítica de las cónicas que es de interés para la mecánica celeste: el área encerrada por estas curvas. Como veremos en el ??, el área es la única cantidad geométrica cuyo valor puede predicirse de forma exacta como función del tiempo cuando describimos el movimiento de un cuerpo respecto a un centro de atracción.

Un poco de historia

Newton, el área debajo de la hipérbola y la invención del cálculo. El cálculo del área encerrada por curvas arbitrarias (cuadratura) es uno de los problemas clásicos de la geometría y en los 1600 uno de los que motivó la invención del cálculo infinitesimal (ver Sección 4.1.7.)

En la Figura 4.21 se muestran las construcciones geométricas que usaremos en las próximas sesiones para calcular el área encerrada por elipses, hipérbolas y

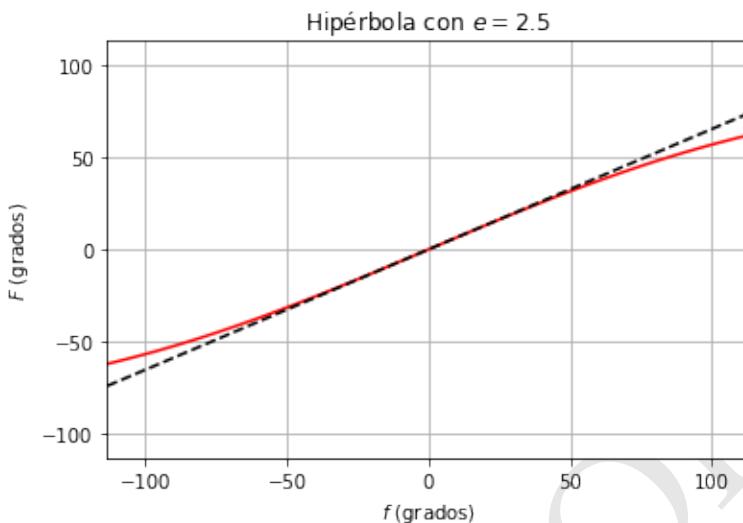


Figura 4.20: Anomalía excéntrica F como función de la anomalía verdadera f para una hipérbola. La línea punteada corresponde a la aproximación $F \approx \sqrt{(e-1)/(e+1)}f$.

paráolas en función de las anomalías definidas en las secciones anteriores y otros parámetros geométricos de esas mismas cónicas.

En los tres casos el área del *sector de cónica* PQF, que es el área de interés para nosotros en la mecánica celeste, se puede siempre escribir como la suma del segmento PQO y del triángulo PFQ:

$$\Delta A \equiv A_{PQF} = A_{PFQ} + A_{PQO} \quad (4.102)$$

El problema, en las tres cónicas consiste en calcular estas dos áreas como función de las anomalías excéntricas (en el caso de la elipse y la hipérbola y de la anomalía verdadera en el caso de la parábola).

Área de un sector de elipse

El área del triángulo en PFQ en la elipse (ver panel superior en [Figura 4.21](#)) se puede escribir en términos de la anomalía excéntrica E como:

$$\begin{aligned} A_{PFQ} &= \frac{1}{2} b \sin E (a \cos E - ae) \\ &= \frac{1}{2} ab \sin E \cos E - abe \sin E \end{aligned} \quad (4.103)$$

Por otro lado, el área del segmento de elipse PQO se puede expresar como la integral definida:

$$A_{PQO} = \int_{-a}^{x_c} y_c(x) dx$$

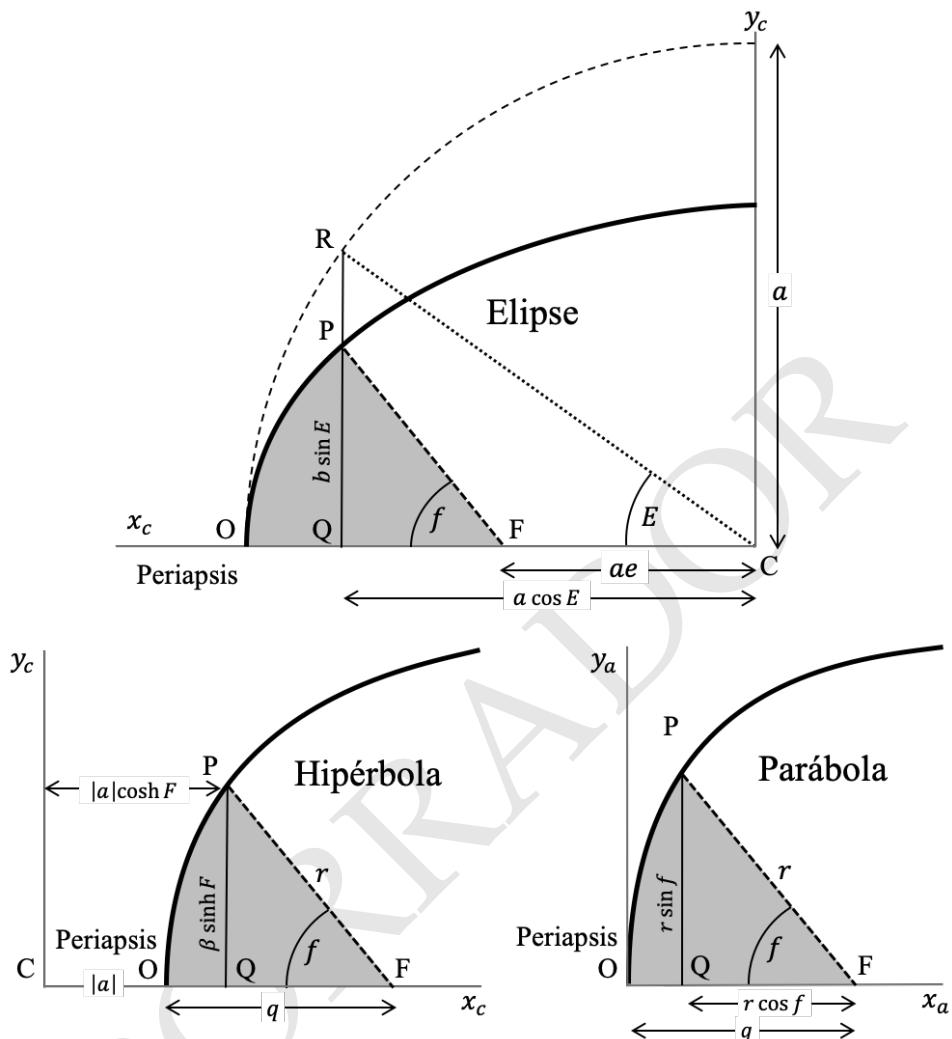


Figura 4.21: Construcción geométrica usada aquí para deducir la relación entre las anomalías f , E y F y el área del sector de cónica (región sombreada).

Por las propiedades geométricas de la elipse (ver [Sección 4.2.14](#)) la ordenada del punto P es $y_c(x) = (b/a)Y(x)$, donde Y es la correspondiente ordenada del punto R sobre la circunferencia circunscrita en la elipse de radio a . De esta manera el área del segmento de elipse se puede expresar en términos del área de segmento de círculo RQO como:

$$A_{PQO} = \frac{1}{2} A_{RQO} \quad (4.104)$$

Por su parte:

$$A_{RQO} = A_{RCO} - A_{RCQ}$$

que a su vez se escriben en términos de E como:

$$A_{RQO} = \frac{1}{2}a^2E - \frac{1}{2}a^2 \sin E \cos E \quad (4.105)$$

Reemplazando la Ec. (4.105) en la (4.104) y esta última junto con la Ec. (4.103) en la fórmula original (Ec. 4.102) obtenemos finalmente:

$$\Delta A_{\text{elipse}} = \frac{1}{2}ab(E - \sin E) \quad (4.106)$$

Siendo la elipse la única figura cerrada entre las cónicas, es posible calcular el área total encerrada por ella haciendo en la última ecuación $E = 2\pi$:

$$A_{\text{elipse}} = \pi abE \quad (4.107)$$

Área de un sector de hipérbola

La construcción en la ?? permite, en el caso de la hipérbola (ver panel inferior izquierdo) escribir las áreas de interés en la Ec. (4.102) como:

$$A_{PFQ} = \frac{1}{2}\beta \sinh F(|a| + q - |a| \cosh F)$$

$$A_{PQO} = \int_{|a|}^{|a| \cosh F} \beta \sqrt{\frac{x^2}{a^2} - 1} dx$$

Para la última expresión usamos la ecuación de la hipérbola referida al centro (ver Ec. 4.78).

Resolviendo la integral (ver problemas al final del capítulo), sumando las dos áreas y teniendo en cuenta que $q = |a|(e - 1)$ obtenemos finalmente el área del sector de hipérbola:

$$\Delta A_{\text{hipérbola}} = \frac{1}{2}|a|\beta(e \sinh F - F) \quad (4.108)$$

Área de un sector de parábola

De forma análoga a como lo hicimos con la hipérbola, podemos escribir las áreas que componen el sector de parábola en términos de la anomalía verdadera f (ver panel inferior derecho en la ??):

$$A_{PFQ} = \frac{1}{2}r^2 \sin f \cos f$$

$$A_{PQO} = \int_0^{q-r \cos f} \sqrt{2px} dx$$

donde en la integral hemos usado la ecuación de la parábola con origen en el ápice $y_a(x_a)^2 = 2px_a$ (ver Ec. 4.72).

A pesar de que estas expresiones parecen más fáciles de desarrollar matemáticamente, en realidad encontrar una versión simplificada del área del sector de

parábola es más complicado de lo que es para el caso del sector de elipse y el de hipérbola.

En términos llanos (sin simplificaciones trigonométricas), el área del sector, que es la suma de las ecuaciones anteriores resulta ser:

$$\Delta A_{\text{parabola}} = \frac{1}{2} \frac{p^2 \sin f \cos f}{(1 + \cos f)^2} + \frac{p^2}{3} \left(\frac{1 - \cos f}{1 + \cos f} \right)^{3/2}$$

donde hemos usado la ecuación de la parábola en coordenadas cilíndricas, $r = p/(1 + \cos f)$ y el hecho que $q = p/2$.

La fracción en el segundo término del lado derecho se puede escribir como:

$$\frac{1 - \cos f}{1 + \cos f} = \tan \frac{f}{2}$$

y esto nos ofrece una clave de cómo simplificar el primer término: escribiendo $\sin f$ y $\cos f$ en términos de $\tan(f/2)$:

$$\sin f = \frac{2 \tan \frac{f}{2}}{1 + \tan^2 \frac{f}{2}} \quad (4.109)$$

$$\cos f = \frac{1 - \tan^2 \frac{f}{2}}{1 + \tan^2 \frac{f}{2}} \quad (4.110)$$

(4.111)

Reemplazando, el área del sector de parábola queda:

$$\Delta A_{\text{parabola}} = \frac{1}{4} p^2 \tan \frac{f}{2} \left(1 - \tan^2 \frac{f}{2} \right) + \frac{p^2}{6} \tan^3 \frac{f}{2}$$

y simplifando obtenemos finalmente:

$$\Delta A_{\text{parabola}} = \frac{1}{4} p^2 \left(\tan \frac{f}{2} + \frac{1}{3} \tan^3 \frac{f}{2} \right) \quad (4.112)$$

4.2.16. Cónicas en el espacio

En las secciones anteriores desarrollamos todas las posibles representaciones algebraicas de las curvas cónicas sobre el plano en el que están definidas. En esta sesión daremos el salto a tres dimensiones y resolveremos la pregunta de ¿cuál es la representación algebraica o geométrica más general de las cónicas en el espacio?

En la [Sección 4.2.9](#) habíamos visto que es posible, partiendo de la descripción algebraica de la cónica en un sistema de coordenadas referido a su eje de simetría y que llamaremos en lo sucesivo el *sistema de referencia natural de la cónica*, aplicar una rotación sobre el plano para obtener la ecuación más general de la cónica sobre ese mismo plano. El salto al espacio de tres dimensiones es simplemente una generalización de este procedimiento.

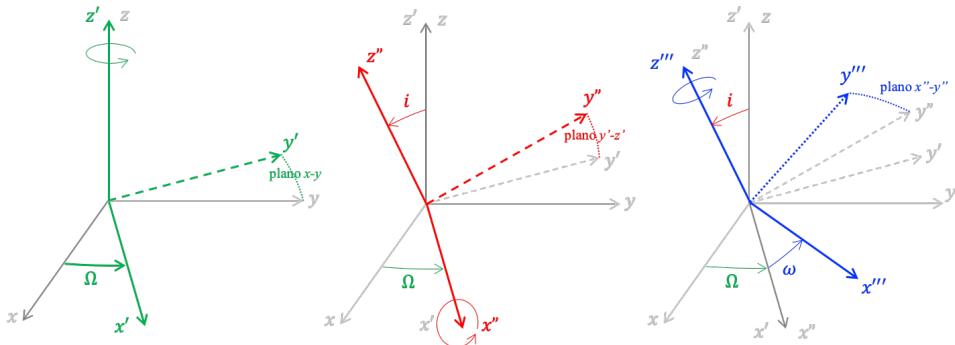


Figura 4.22: Secuencia de rotaciones que permiten pasar del sistema natural de ejes de la cónica $x - y - z$ a un sistema con una orientación arbitraria $x''' - y''' - z'''$

4.2.17. Ángulos de Euler

Partiendo del sistema de referencia natural de la cónica, es posible orientar de forma arbitraria la curva en el espacio realizando en total tres rotaciones *independientes* (ver Figura 4.22.) Los ángulos en los que se realizan esas rotaciones, y que llamaremos en este texto (Ω, i, ω) , se conocen universalmente como los **ángulos de Euler**.

La secuencia de rotaciones mostradas en la ?? se puede describir cualitativamente como:

1. Rotación del sistema original $x - y - z$ (sistema natural) en un ángulo Ω alrededor del eje z , para obtener un nuevo sistema de ejes $x' - y' - z'$
2. Rotación del sistema $x' - y' - z'$ en un ángulo i alrededor del eje z' , para obtener un nuevo sistema de ejes $x'' - y'' - z''$
3. Rotación del sistema $x'' - y'' - z''$ en un ángulo ω alrededor del eje z'' , para obtener un nuevo sistema de ejes $x''' - y''' - z'''$.

Al sistema de ejes final lo llamamos el *sistema de coordenadas del observador*.

Usando la representación matricial de las rotaciones en el plano de la Ec. (4.82), la relación entre las coordenadas del observador y las coordenadas naturales de las cónicas será:

$$\begin{pmatrix} x''' \\ y''' \\ z''' \end{pmatrix} = R_z(\omega)R_x(i)R_z(\Omega) \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (4.113)$$

Si usamos la definición de las matrices de rotación de la Ec. (4.83):

$$\begin{pmatrix} x''' \\ y''' \\ z''' \end{pmatrix} = \begin{pmatrix} c\omega & s\omega & 0 \\ -s\omega & c\omega & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & ci & si \\ 0 & -si & ci \end{pmatrix} \begin{pmatrix} c\Omega & s\Omega & 0 \\ -s\Omega & c\Omega & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

donde se ha abreviado $c\theta \equiv \cos \theta$ y $s\theta \equiv \sin \theta$.

Al realizar las multiplicaciones matriciales explícitas queda:

$$\begin{pmatrix} x''' \\ y''' \\ z''' \end{pmatrix} = \mathcal{M}(\Omega, i, \omega) \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (4.114)$$

donde la matriz de rotación en tres dimensiones,

$$\mathcal{M}(\omega, i, \Omega) \equiv R_z(\omega)R_x(i)R_z(\Omega) \quad (4.115)$$

se escribe explícitamente como:

$$\mathcal{M}(\omega, i, \Omega) = \begin{pmatrix} c\omega c\Omega - ci s\omega s\Omega & c\omega s\Omega + c\Omega ci s\omega & si s\omega \\ -c\Omega s\omega - s\Omega ci c\omega & -s\Omega s\omega + c\Omega ci c\omega & si c\Omega \\ s\Omega si & -c\Omega si & ci \end{pmatrix} \quad (4.116)$$

Por ser \mathcal{M} el producto de matrices de unitarias (Ec. 4.115), ella es en sí misma una matriz unitaria, es decir $\det \mathcal{M} = 1$, pero más importante:

$$\mathcal{M}^{-1} = \mathcal{M}^T$$

explícitamente,

$$\mathcal{M}(\omega, i, \Omega)^T = \begin{pmatrix} c\omega c\Omega - ci s\Omega s\omega & -c\Omega s\omega - c\omega ci s\Omega & si s\Omega \\ c\omega s\Omega + s\omega ci c\Omega & -s\Omega s\omega + c\omega ci c\Omega & si c\omega \\ s\omega si & c\omega si & ci \end{pmatrix} \quad (4.117)$$

A la inversa, las coordenadas naturales de la cónica (x, y, z) se pueden obtener de las coordenadas del observador, invirtiendo la Ec. (??):

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = R_z(-\Omega)R_x(-i)R_y(-\omega) \begin{pmatrix} x''' \\ y''' \\ z''' \end{pmatrix} \quad (4.118)$$

o equivalentemente, usando la forma explícita en la Ec. (4.114) y aprovechando las propiedades de la matriz de rotación en tres dimensiones \mathcal{M} :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathcal{M}^T \begin{pmatrix} x''' \\ y''' \\ z''' \end{pmatrix} \quad (4.119)$$

4.2.18. Matrices de rotación generales

Usando SPICE podemos construir la matriz de rotación en tres dimensiones \mathcal{M} por dos medios distintos. El primero es usar la rutina `rotate` que habíamos introducido en el Alg. (4.13):

```
#Angulos
from numpy import pi
Omega=pi/6
```

```
omega=pi/3
i=pi/4

#Matrices individuales
from spiceypy import rotate
RzOmega=rotate(Omega,3)
Rxi=rotate(i,1)
Rzomega=rotate(omega,3)
```

La matriz de rotación en tres dimensiones \mathcal{M} , de acuerdo con su definición en la Ec. (4.115), puede obtenerse aplicando sucesivamente la rutina de multiplicación de matrices `mxm`:

(Algoritmo 4.21)

```
from spiceypy import mxm
M=mxm(Rzomega,mxm(Rxi,RzOmega))
```

```
M =
[[ 0.127  0.78   0.612]
 [-0.927 -0.127  0.354]
 [ 0.354 -0.612  0.707]]
```

Existe sin embargo una rutina compacta y mucho más general en SPICE que permite calcular la matriz rotación de cualquier sucesión de rotaciones en el espacio, no solamente la que usamos aquí.

Para ello podemos definir una matriz de rotación general \mathcal{R} resultante de aplicar rotaciones sucesivas θ_i alrededor del eje \hat{e}_k , θ_j alrededor del eje \hat{e}_j y θ_i alrededor del eje \hat{e}_i como:

$$\mathcal{R}(\theta_i, \theta_j, \theta_k, i, j, k) \equiv R_i(\theta_i)R_j(\theta_j)R_k(\theta_k)$$

En términos de esta matriz general, la matriz de rotación *canónica* \mathcal{M} presentada en la Sección 4.2.17 se escribe:

$$\mathcal{M} = \mathcal{R}(\omega, i, \Omega, 3, 1, 3)$$

En el paquete SPICE la rotación general \mathcal{R} se implementa con la rutina `eul2m` que podemos usar para obtener \mathcal{M} como:

(Algoritmo 4.22)

```
from spiceypy import eul2m
M=eul2m(omega,i,Omega,3,1,3)
```

```
M =
[[ 0.127  0.78   0.612]
 [-0.927 -0.127  0.354]
 [ 0.354 -0.612  0.707]]
```

Naturalmente el resultado coincide con el obtenido en el Alg. (4.21). Podemos ahora verificar la unitariedad de \mathcal{M} calculando su determinante, su inversa (calculada numérica y usando la propiedad en la Ec. 4.118) y su transpuesta:

```

#Determinante
from numpy.linalg import det
detM=det(M)

#Inversa
from numpy.linalg import inv
Minv=inv(M)

#Inversa por definicion
Minv_def=eul2m(-Omega,-i,-omega,3,1,3)

#Transpuesta
MT=M.transpose()

print(f"det(M) = {detM:g}")
print(f"inversa M (numérica) = \n{Minv}")
print(f"inversa M (definición) = \n{Minv_def}")
print(f"transpuesta M = \n{MT}")

det(M) = 1
inversa M (numérica) =
[[ 0.127 -0.927  0.354]
 [ 0.78   -0.127 -0.612]
 [ 0.612  0.354  0.707]]
inversa M (definición) =
[[ 0.127 -0.927  0.354]
 [ 0.78   -0.127 -0.612]
 [ 0.612  0.354  0.707]]
transpuesta M =
[[ 0.127 -0.927  0.354]
 [ 0.78   -0.127 -0.612]
 [ 0.612  0.354  0.707]]

```

4.2.19. Gráfico de una cónica rotada en el espacio

Con estos elementos a la mano podemos escribir algoritmos para, usando las ecuaciones encontradas en la Sección 4.2.11 y los algoritmos de la Sección 4.2.11, representar gráficamente una cónica en el espacio.

Comenzamos por obtener los puntos de la cónica en su sistema natural de coordenadas (plano de la cónica, origen en el foco, semieje $x+$ apuntando hacia el perihelio) usando para ello la rutina `puntos_conica` que habíamos introducido en el Alg. (4.19):

```

from pymcel.export import puntos_conica
p=10.0
e=0.8
xs,ys,zs=puntos_conica(p,e)

```

Ahora podemos rotar los puntos de la cónica usando la matriz M calculada en el Alg. (4.22). Para hacerlo nos apoyaremos nuevamente en la rutina general

`rota_puntos` que habíamos introducido antes en el Alg. (??):

```
from pymcel.export import rota_puntos
xppps,yppps,zppps=rota_puntos(M, xs, ys, zs)
```

Una gráfica en tres dimensiones de la cónica en su plano natural y en los ejes rotados se puede realizar usando el siguiente código:

(Algoritmo 4.23)

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure()
ax=fig.gca(projection='3d')

#Gráfica de los puntos originales
ax.plot(xs,ys,zs, 'k--')

#Gráfica de la cónica rotada
ax.plot(xppps,yppps,zppps, 'b-')

#Decoración
ax.set_xlabel("$x$")
ax.set_ylabel("$y$")
ax.set_zlabel("$z$")

from pymcel.plot import fija_ejes3d_proporcionales
fija_ejes3d_proporcionales(ax);
fig.tight_layout();
```

ver Figura 4.23

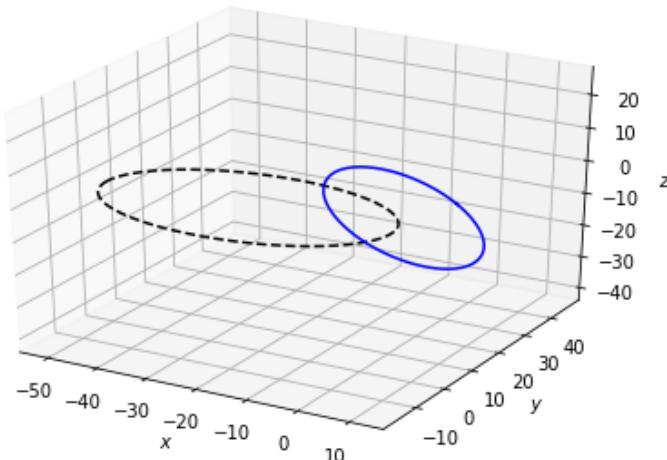


Figura 4.23: Figura correspondiente al código 4.23.

4.2.20. Elementos orbitales

Partiendo de las ecuaciones paramétricas de la cónica en su plano natural (Ecs. 4.94 y 4.95):

$$\begin{aligned}x''' &= \frac{p \cos f}{1 + e \cos f} \\y''' &= \frac{p \sin f}{1 + e \cos f} \\z''' &= 0\end{aligned}$$

y usando las expresiones explícitas para la rotación en tres dimensiones dadas por las Ecs. (4.119) y (4.117), podemos escribir las ecuaciones paramétricas generales de una cónica en el espacio:

$$\begin{aligned}x &= r[\cos \Omega \cos(\omega + f) - \cos i \sin \Omega \sin(\omega + f)] \\y &= r[\sin \Omega \cos(\omega + f) + \cos i \cos \Omega \sin(\omega + f)] \\z &= r[\cos f \sin \omega \sin i + \sin f \cos \omega \sin i]\end{aligned}\quad (4.120)$$

donde $r = p / (1 + e \cos f)$

Vemos aquí entonces que para especificar la posición de cualquier punto sobre una cónica, independiente de su orientación espacial, hace falta indicar el valor de 6 parámetros: p, e, i, Ω, ω y f . A estas cantidades las llamamos en mecánica celeste los **elementos orbitales clásicos** y volveremos sobre ellos en el ??.

En realidad, de los 6 elementos orbitales clásicos, 5 de ellos (p, e, i, Ω, ω) permiten especificar el tamaño, forma y orientación de la cónica y son compartidos por todos los puntos que definen la curva. El último elemento, f permite especificar la posición de un punto específico.

Usando los elementos orbitales clásicos podemos dibujar una cónica en el espacio usando un algoritmo más directo que el que usamos en la Sección 4.2.19. Para ello hemos diseñado la rutina `conica_de_elementos`:

(Algoritmo 4.24)

```
def conica_de_elementos(p=10.0,e=0.8,i=0.0,Omega=0.0,omega=0.0,
                        df=0.1,
                        elev=30,azim=60,
                        figreturn=False):

    #Convierte elementos angulares en radianes
    from numpy import pi
    p=float(p)
    e=float(e)
    i=float(i)*pi/180
    Omega=float(Omega)*pi/180
    omega=float(omega)*pi/180

    #Compute fmin,fmax
    if e<1:
        fmin=-pi
        fmax=pi
```

```
elif e>1:
    from numpy import arccos
    psi=arccos(1/e)
    fmin=-pi+psi+df
    fmax=pi-psi-df
else:
    fmin=-pi+df
    fmax=pi-df

#Valores del ángulo
from numpy import linspace,pi
fs=linspace(fmin,fmax,500)

#Distancia al periapsis
q=p/(1+e)

#Distancia al foco
from numpy import sin,cos
rs=p/(1+e*cos(fs))

#Coordinadas
xs=rs*(cos(Omega)*cos(omega+fs)-cos(i)*sin(Omega)*sin(omega+fs))
ys=rs*(sin(Omega)*cos(omega+fs)+cos(i)*cos(Omega)*sin(omega+fs))
zs=rs*(cos(fs)*sin(omega)*sin(i)+sin(fs)*cos(omega)*sin(i))

#Posición del periapsis (f=0)
xp=q*(cos(Omega)*cos(omega)-cos(i)*sin(Omega)*sin(omega))
yp=q*(sin(Omega)*cos(omega)+cos(i)*cos(Omega)*sin(omega))
zp=q*sin(omega)*sin(i)

#Posición del nodo ascendente
rn=p/(1+e*cos(omega))
xn=rn*cos(Omega)
yn=rn*sin(Omega)
zn=0

#Gráfico
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
plt.close("all")
fig=plt.figure()
ax=fig.gca(projection='3d')

#Gráfica de los puntos originales
ax.plot(xs,ys,zs,'b-')

#Posición del periapsis
ax.plot([0,xp],[0,yp],[0,zp],'r-')

#Posición del nodo ascendente
ax.plot([0,xn],[0,yn],[0,zn],'g-')
```

```

#Fija punto de vista
ax.view_init(elev=elev,azim=azim)

#Decoración
from pymcel.plot import fija_ejes3d_proporcionales
xrange,yrange,zrange=fija_ejes3d_proporcionales(ax);

ax.set_title(f"Cónica con:{p:.2f}, {e:.2f}, "+\
            f"{i={i*180/pi:.2f}}, "+\
            f"${\Omega}ega={\Omega}ega*180/pi:.1f}, "+\
            f"${\omega}ega={\Omega}ega*180/pi:.1f}"
)

#Dibuja Ejes
ax.plot([0,xrange[1]],[0,0],[0,0],'k-')
ax.plot([0,0],[0,yrange[1]],[0,0],'k-')
ax.plot([0,0],[0,0],[0,zrange[1]],'k-')
ax.text(xrange[1],0,0,"$x$",ha='left',va='top')
ax.text(0,yrange[1],0,"$y$",ha='left',va='top')
ax.text(0,0,zrange[1],"$z$",ha='left',va='bottom')

fig.tight_layout();

if figreturn: return fig

```

El gráfico de la cónica sería:

(Algoritmo 4.25)

```
from numpy import pi
conica_de_elementos(p=10.0,e=0.5,Omega=60.0,i=30.0,omega=20.0);
```

ver Figura 4.24

Para ver una versión interactiva de esta gráfica vaya a las libertas disponibles en la [versión electrónica el libro²⁶](#).

²⁶<http://github.com/seap-udea/MecanicaCeleste-Zuluaga>

Cónica con: $p = 10.00$, $e = 0.50$, $i = 30.00$, $\Omega = 60.0$, $\omega = 60.0$

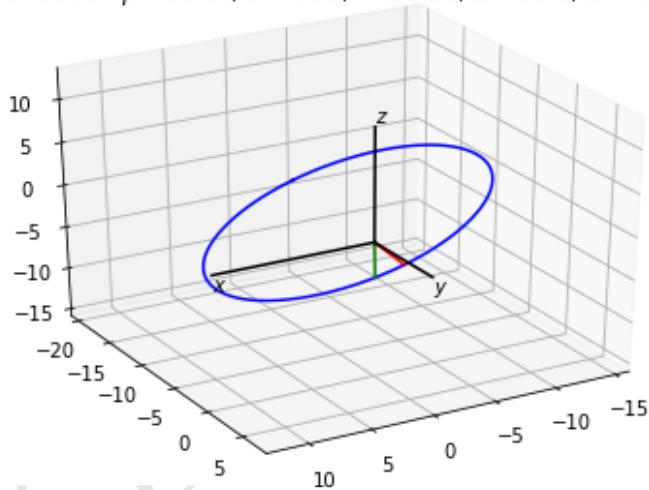


Figura 4.24: Figura correspondiente al código 4.25.

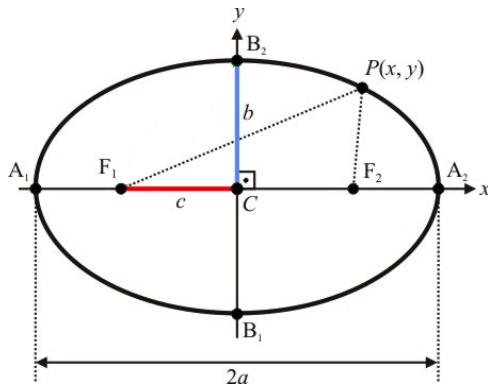


Figura 4.25: Una elipse con algunos puntos resaltados.

4.3. Problemas seleccionados

1. **Otra definición de una elipse.** Una elipse se puede definir como el lugar geométrico de los puntos cuya suma de las distancias a dos puntos fijos llamados focos es constante (esto es, en la figura de abajo, que $\overline{F_1P} + \overline{F_2P} = \text{constante}$). A partir de esta definición y de los parámetros descritos en el punto anterior:

- Muestre que la constante a la que es igual dicha suma es $2a$.
- Muestre que a , b y la *distancia focal* —definida como la distancia entre el centro C de la elipse y cualquiera de sus focos—, c , satisfacen el teorema de pitágoras $a^2 = b^2 + c^2$.
- Demuestre a partir de esta definición que la ecuación de la elipse horizontal centrada en $C = (0,0)$ es:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

- A partir de las relaciones dadas entre a , b , e y F , muestre que e también se puede calcular como $e = c/a$.
 - A partir de la definición conceptual de p dada en el punto 1, muestre que el *semilatus rectum* también se puede calcular como $p = b^2/a$.
2. **Ecuación de una elipse.** Encuentre la ecuación general de la elipse cuyo semilatus rectum valga 9 y excentricidad sea 0.5.
3. **Parámetros geométricos.** Defina conceptualmente a qué corresponden cada uno de los siguientes parámetros en el caso de una elipse: a) defina F , b) defina e , c) defina p , d) defina C , e) defina a , f) defina b , g) defina r , h) defina f , i) defina E .

- 4. Traslación de las cónicas.** A partir de una completación de cuadrados, demuestre que la ecuación general de las cónicas sin términos acoplados

$$Ax^2 + Cy^2 + Dx + Ey + G = 0$$

con $A, C \neq 0$ se puede escribir, en su forma ordinaria, como

$$\frac{x'^2}{q} + \frac{y'^2}{t} = 1,$$

donde x' y y' son las coordenadas de un punto en el sistema trasladado dadas por

$$\begin{aligned}x' &= x + \frac{D}{2A}, \\y' &= y + \frac{E}{2C}\end{aligned}$$

y q, t están dadas por

$$\begin{aligned}q &= -\frac{G}{A} + \frac{D^2}{4A^2} + \frac{E^2}{4AC}, \\t &= -\frac{G}{C} + \frac{D^2}{4AC} + \frac{E^2}{4C^2}.\end{aligned}$$

¿Qué condiciones se deben dar para que dicha cónica sea una circunferencia, una elipse o una hipérbola? Dé un ejemplo numérico sencillo de cada una.

- 5. Rotación de cónicas.** A partir de la ecuación general de la cónica con términos acoplados

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + G = 0$$

aplique una rotación de la forma

$$\begin{aligned}x &= x' \cos \theta - y' \sin \theta \\y &= x' \sin \theta + y' \cos \theta\end{aligned}$$

y demuestre que si se rota el sistema de coordenadas original un ángulo θ que satisface

$$\tan 2\theta = \frac{B}{A - C},$$

los nuevos ejes estarán alineados con los ejes de simetría.

6. **Órbitas de cometas.** El semieje mayor de un cometa es de 20 AU y la distancia entre el foco y el perihelio es de 1 AU.

- ¿Cuál es la excentricidad de la órbita?
- ¿Cuánto vale el semieje menor?
- ¿Cuánto vale el semilatus rectum?
- Escriba la ecuación de este cometa en coordenadas polares.
- Haga un bosquejo de la situación medianamente a escala.

7. **Órbita lunar.** Sobre una órbita lunar, el punto más cercano al satélite se conoce como periselenio y el más lejano como aposelenio. La nave Apolo 11 fue puesta en una órbita elíptica alrededor de la luna con una altitud (respecto a la superficie lunar) en el periselenio de 110 km y de 314 km en el aposelenio. Encuentre la ecuación de esta elipse en coordenadas polares si el radio de la luna es de 1728 km y el centro de la luna se encuentra en uno de los focos de la elipse. Haga un bosquejo de la situación medianamente a escala.

8. Anomalías.

- Demuestre que, para una elipse, la distancia radial, r , está relacionada con la anomalía excéntrica, E , por medio de la expresión

$$r = a(1 - e \cos E).$$

- Demuestre, además, que la anomalía verdadera se relaciona con la anomalía excéntrica a través de

$$\tan \frac{f}{2} = \sqrt{\frac{1+e}{1-e}} \tan \frac{E}{2}.$$

9. **Parámetros de elipse.** Dada la ecuación de la elipse $x^2 + 2y^2 + 2x + y - 6 = 0$ complete cuadrados y llévela a su forma ordinaria, para determinar la posición del centro, el semieje mayor, semieje menor, distancia focal y semilatus rectum.

10. **Elementos orbitales.** Para un satélite en órbita alrededor de la Tierra se tienen los siguientes elementos orbitales: $a = 8016,0$ km, $e = 0,06$, $I = 50^\circ$, $\Omega = 0,0^\circ$, $\omega = 30^\circ$ y $f = 20^\circ$. Encuentre el vector de posición en el plano fundamental del ecuador terrestre.

11. **Circunferencia máxima.** La posición de un punto sobre la Tierra se especifica por su longitud ϕ y latitud λ , como se muestra en la [Figura 4.26](#).

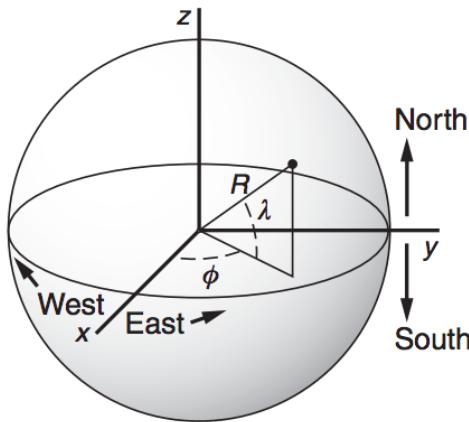


Figura 4.26: Definición de las coordenadas de latitud y longitud sobre la Tierra

Sean los vectores desde el centro de la Tierra hacia dos puntos \vec{r}_1 y \vec{r}_2 . El coseno del ángulo θ entre ellos puede ser hallado a partir de su producto escalar (producto punto), de tal manera que la distancia a lo largo del círculo máximo entre los dos puntos es $R\theta$.

Encuentre una expresión para θ en términos de las coordenadas de los dos puntos.

12. **Derivada temporal de los vectores unitarios.** La figura muestra la configuración de un sistema de coordenadas polares definido por $(\hat{r}, \hat{\theta})$ relativo a un sistema cartesiano. Los conjuntos de vectores $(\hat{r}, \hat{\theta})$ y (\hat{i}, \hat{j}) son constantes en magnitud pero solamente los vectores cartesianos lo son también en dirección. Conforme la partícula de vector posición \vec{r} se desplaza, los vectores $(\hat{r}, \hat{\theta})$ cambian de dirección, de forma tal que es posible definir una derivada temporal de estos.

- a) Muestre que

$$\frac{d\hat{r}}{dt} = \dot{\theta}\hat{\theta}$$

$$\frac{d\hat{\theta}}{dt} = -\dot{\theta}\hat{r}$$

- b) Usando los resultados del punto anterior, muestre que el vector aceleración en coordenadas polares viene dado por

$$\vec{a} = (\ddot{r} - r\dot{\theta}^2)\hat{r} + (r\ddot{\theta} + 2\dot{r}\dot{\theta})\hat{\theta}$$

Bibliografía

- [1] T. M. APOSTOL, *CALCULUS volume I One-Variable Calculus, with an Introduction to Linear Algebra*, Blaisdell Publishing Company, John Wiley & Sons, 1967.
- [2] T. M. APOSTOL, *Calculus, Volume II: Multi-Variable Calculus and Linear Algebra, with Applications to Differential Equations and Probability*, John Wiley & Sons, 1969.
- [3] K. BATYGIN AND M. E. BROWN, *Evidence for a distant giant planet in the solar system*, The Astronomical Journal, 151 (2016), p. 22.
- [4] H. W. EVES, *A Survey of Geometry: Rev. Ed*, Allyn and Bacon, 1972.
- [5] A. C. HINDMARSH, *Odepack, a systematized collection of ode solvers*, Scientific computing, (1983), pp. 55–64.
- [6] I. NEWTON AND E. HALLEY, *Philosophiae naturalis principia mathematica*, vol. 62, Jussu Societatis Regiae ac typis Josephi Streator, prostant venales apud Sam ..., 1780.
- [7] W. H. PRESS, S. A. TEUKOLSKY, W. T. VETTERLING, AND B. P. FLANNERY, *Numerical recipes 3rd edition: The art of scientific computing*, Cambridge university press, 2007.